AD-A053 336 INTEGRATED SCIENCES CORP SANTA MONICA CALIF F/G 5/8
EXPERIMENTAL INVESTIGATION OF THE USEFULNESS OF OPERATOR AIDED --ETC(U)
JAN 78 D H WALSH, M D SCHECHTERMAN N00014-75-C-0811
UNCLASSIFIED ISC-215-4 NL

1 OF 2
AD
A053336

2 (12)

EXPERIMENTAL INVESTIGATION OF THE USEFULNESS OF OPERATOR AIDED OPTIMIZATION IN A SIMULATED TACTICAL DECISION AIDING TASK

REPORT NO. 215-4

PREPARED FOR:

DIRECTOR, ENGINEERING PSYCHOLOGY PROGRAM
CODE 455
PSYCHOLOGICAL SCIENCES DIVISION
OFFICE OF NAVAL RESEARCH
DEPARTMENT OF THE NAVY
ARLINGTON, VIRGINIA 22217

JANUARY 1978

INTEGRATED SCIENCES CORPORATION
Santa Monica, California

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (*and Subtitle*) EXPERIMENTAL INVESTIGATION OF THE USEFULNESS OF OPERATOR AIDED OPTIMIZATION IN A SIMULATED TACTICAL DECISION AIDING TASK. 6		5. TYPE OF REPORT & PERIOD COVERED 9 TECHNICAL REPORT,
		6. PERFORMING ORG. REPORT NUMBER 14 ISC-215-4
7. AUTHOR(s) 10 David H. WALSH, Michael D. SCHECHTERMAN		8. CONTRACT OR GRANT NUMBER(s) 15 N00014-75-C-0811
9. PERFORMING ORGANIZATION NAME AND ADDRESS INTEGRATED SCIENCES CORPORATION 1640 Fifth Street, Suite 204 Santa Monica, California 90401		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS ENGINEERING PSYCHOLOGY PROGRAM, CODE 455 OFFICE OF NAVAL RESEARCH ARLINGTON, VIRGINIA 22217		12. REPORT DATE 11 JAN 1978
		13. NUMBER OF PAGES 163 12 178
14. MONITORING AGENCY NAME & ADDRESS(*if different from Controlling Office*)		15. SECURITY CLASS. (*of this report*) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (*of this Report*)

UNLIMITED

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

DDC
RECEIVED
MAY 1 1978
A

17. DISTRIBUTION STATEMENT (*of the abstract entered in Block 20, if different from Report*)

18. SUPPLEMENTARY NOTES

Reproduction in whole or in part is permitted for any purpose of the United States Government.

ORIGINAL CONTAINS COLOR PLATES: ALL DDC REPRODUCTIONS WILL BE IN BLACK AND WHITE

19. KEY WORDS (*Continue on reverse side if necessary and identify by block number*)

Decision Aids	Dynamic Programming Optimization
Naval Tactics	Command and Control
Interactive Graphics	Mission Planning
Man-Computer Functional Allocation	
Nonlinear Programming Optimization	

20. ABSTRACT (*Continue on reverse side if necessary and identify by block number*)

This is a report of work accomplished on one task in the project entitled, Operational Decision Aids, which was initiated in 1974 by the Office of Naval Research to develop aids for Navy command and control functions and make them available for incorporation in the design of future systems such as the Tactical Flag Command Center (TFCC). The report describes the implementation and experimental evaluation of two examples of a decision aid for Naval Task Force Commanders called "operator aided optimization" (OAO). This type of aid

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

404 569

20. Abstract (Continued)

enables the user to make inputs to a machine which operates on the inputs and shows problem solutions derived from the inputs. The user and the machine work iteratively until the user is satisfied with the machine solution. Problems for which OAO is appropriate tend to have the following characteristics:

1. Solution space is of high dimensionality ~~(e.g., >5)~~;
2. Criterion function is nonlinear, and multi-modal, therefore the machine working alone may only find a local optimum instead of the global optimum.

The problem for which the ISC aids were constructed is the selection of (a) an air strike path through a field of ten enemy sensors and (b) aircraft speeds on each leg of the path. Utility of each candidate strike path was computed according to a predetermined utility function; the utility function was nonlinear and multi-modal. One of the ISC-designed aids uses a nonlinear programming (NP) algorithm; the other uses a dynamic programming (DP) algorithm. An experiment was designed by ISC to compare decision performance for both types of aids. Sixteen UCLA students solved problems with and without the aid and these data were compared. Subject data using the aids were also compared with the data resulting from using the algorithm in the automated mode. The principal findings from the experiment were:

1. The operators using the NP aid did significantly better than without the aid. The average improvement across all subjects and trials was 29% with a range of 9% to 123%. Performance was significantly different across operators but this was solely for unaided operation. Thus the aid served as an "equalizer." It enabled operators having relatively low scores without the aid to do as well as those who had relatively high scores without the aid.

2. Operators using the DP aid did significantly better than without the aid. The average improvement across all subjects and trials was 12% with a range of 3.5% to 27%.

3. The lack of a technical education was apparently not an impediment to good performance with or without either aid.

4. The average time required to adequately train an operator to use either aid was about four hours.

5. A potential implication of the findings is that OAO is attractive to use when it is applicable because:

   a. The operator can see what is happening during the optimization. With pictorial problem representation, he can make adjustments to the optimization process or results to compensate for limitations in problem representation more easily than when there is no pictorial representation.

   b. The time required to train operators to use OAO with pictorial problem representation is apparently relatively short and does not require technical knowledge of the optimization algorithms.

EXPERIMENTAL INVESTIGATION OF THE USEFULNESS OF OPERATOR AIDED OPTIMIZATION IN A SIMULATED TACTICAL DECISION AIDING TASK

Report No. 215-4

Contract No. N00014-75-C-0811

Prepared for:

Code 455
Director, Engineering Psychology Program
Psychological Sciences Division
Office of Naval Research
Department of the Navy
Arlington, Virginia 22217

By:

David H. Walsh
Michael D. Schechterman

Integrated Sciences Corporation
1640 Fifth Street
Santa Monica, California 90401

January 1978

EXECUTIVE BRIEF

A. BACKGROUND

This is the second report by Integrated Sciences Corporation (ISC) as one of a group of contractors working on the Operational Decision Aids (ODA) program directed by the Office of Naval Research. The ODA program was initiated in 1974. It is intended to develop a variety of decision aids and test and evaluate their usefulness to the Navy. Although the program is not tied to any specific command and control hardware system, it has focused on the functions of a Task Force Commander (TFC) and his staff. The role of ISC has been to find ways to improve man-machine communication by allocating functions between man and machine that take advantage of their respective strengths.

ISC's early work on the ODA program explored the use of techniques by which a decision maker might express and communicate his perception of important relationships. ISC calls the particular techniques it has been developing "Sketch Models." A Sketch Model is essentially a "picture" that is first mentally visualized, and then drawn by a decision maker. As used here, the picture represents the decision maker's perception of the functional relationship between two or more variables, with the stipulation that the function be continuous in at least one dimension. Depending on the application, a Sketch Model can be, for instance, a single curve defining the relationship between two variables, or it can be a family of parameterized curves, or it can be a two-dimensional projection of the iso-"altitude" contours of a three dimensional function.

ISC's first investigation evaluated the ability of human operators to generate Sketch Models of bivariate Gaussian density functions from sampled data. In an experiment, a group of subjects were found capable of developing accurate Sketch Models of one type of well-behaved (i.e., unimodal and symmetric) three-dimensional function. These Sketch Models developed by the subjects from small samples of the underlying functions estimated those functions at least as well as, and in some cases better than, the statistical technique of maximum likelihood estimation.

A second study was undertaken to extend those results by investigating the ability of human operators to generate Sketch Models of less well-behaved functions, i.e., multimodal and unsymmetric. Experimental results gave a very strong indication that the subjects were able to produce accurate (as measured by percent volume error) Sketch Models for a highly irregular (multimodal and unsymmetric) function representing the joint detection capability of multiple sensors (Reference 1).

The study also sought to evaluate the usefulness of those Sketch Models for a decision task represented by an air strike path optimization problem. The experimental results did not prove or disprove the usefulness of the Sketch Model technique as an aid to the decision problem. There were two reasons. One was insufficient data, due to severe attrition in the pool of subjects trained in Sketch Modeling during a lengthy hiatus caused by a series of hardware malfunctions. The second problem surfaced as soon as the data were analyzed: it appears that the problems were too easy for the subjects. It was therefore difficult to ascertain the usefulness of Sketch Models as an aid to the strike path selection problem.

In this study ISC suggests that machine participation in decision making can usefully be examined by distinguishing three types. These types correspond to different degrees of machine participation in decision making. The first type is unaided, or "man-only," decision making. It is relied upon when automated aids either do not exist or are inappropriate to the user's needs or preferences. The second type is machine aided decision making or "machine helping man." Aids of this type have proliferated due to rapid advancements in computer capabilities and operations research. Examples of the machine helping man, as the term is used here, include the many uses of computers to solve product mix problems by linear programming methods and solution of transportation network problems by dynamic programming.

ISC believes that its characterization of decision aiding types can help to distinguish actual TFC decision environments appropriate for each type. In particular, ISC believes that there are a large number of decision environments for which the second type is not useful. This is because the methods available are often too slow, require too much hardware capability, and give questionable answers in difficult environments. This defeats their cost-effectiveness certainly for at-sea use, and quite likely land based planning use, by TFC's.

It is because of these characteristics that ISC suggests and concentrates in this report on defining a third type called "man helping machine to help man." This type of decision aiding finds favor when machine aided decision making is inadequate and one or both of the following circumstances apply:

1. The decision maker does not have a good understanding of how the machine works to obtain a solution. This is often the case when an iterative optimization procedure is used.
2. The decision maker believes that the model used by the machine is not adequately representative of the real world and therefore he wishes to be able to use his knowledge of the real world to compensate for the model's limitation.

B. THE CURRENT STUDY

ISC's previous work on the ODA program had established that humans were adept at perceiving and sketching complex functional relationships when data that could be used to estimate the function were presented to the human in geometric/graphical format. The question became the following: How useful is this human capability? Therefore ISC proceeded to define (a) two decision aids that would use the human capability to solve an experimental problem that could also be solved by a fully automated algorithm, i.e., machine aided decision making and (b) an experiment that would compare decision performance with and without the aids. The two ISC-designed aids were called Operator Aided Optimization (OAO) using Nonlinear Programming (NP) and Operator Aided Optimization using Dynamic Programming (DP). Several key assumptions were made:

1. Some of the complex problems requiring decisions by the Task Force Commander can be treated by analytic methods.

2. Special purpose algorithms could be constructed to solve <u>any</u> problem, but there are so many variations on problems that it would be impossible or uneconomic to have a special purpose algorithm to solve <u>every</u> problem. Therefore, it is likely that the TFC will have a general purpose algorithm available to solve each generic class of problems.

3. The Task Force Commander of the 1980's will have a general purpose computer and computer driven display at his disposal.

It is important to understand the nature and purpose of the experiments reported in this third ISC study for the ODA program. Although ISC used much of the structure and characteristics of a real-world situation, the experiment was deliberately limited and therefore, in a sense, artificial. The problem situation used in the experiment is the selection of (a) an air strike path through a field of ten enemy sensors and (b) aircraft speeds on each leg of the path. (Hereafter in this report, the selection of path and speeds is abbreviated to "selection of path.") Many aspects of real-world air strike planning were not included in the experimental problem, e.g., aircraft altitude, specific locations of enemy weapon systems and such real-world systems as electronic countermeasures. Also, the design of the experimental problems made certain perfect-information assumptions in order to simplify the analysis.

The experiment was principally designed to contrast decision performance obtained with aids of the second and third types. A general purpose algorithm is available to solve the experimental problem in the second type or "machine helping man" mode of aided decision making. However, a "man helping machine to help man" aid which uses man's ability to visually perceive complex functional relationships is also available. This aid uses the same general purpose algorithm as is used in the "machine helping man" case. However, the man now controls the use of the algorithm instead of letting it run open-loop. The experiment designed by ISC compares decision performance for both types of aids.

C. PRINCIPAL FINDINGS

1. The operators using the NP aid did significantly better than without the aid. The average improvement across all subjects and trials was 29% with a range of 9% to 123%. Performance was significantly different across operators but this was solely for unaided operation. Thus the aid served as an "equalizer." It enabled operators having relatively low scores without the aid to do as well as those who had relatively high scores without the aid.

2. Operators using the DP aid did significantly better than without the aid. The average improvement across all subjects and trials was 12% with a range of 3.5% to 27%.

3. The lack of a technical education was apparently not an impediment to good performance with or without either aid.

4. Operator aided optimization was significantly better than automated use of the NP algorithm for both types of rules used by the algorithm to select starting points.

5. The NP aid was less complex to use than the DP aid and operators generally preferred working with the NP aid to working with the DP aid. Operators using OAO with the NP aid found the global optimum on a higher percentage of trials than operators using OAO with the DP aid. The average time required to adequately train an operator to use either aid was about four hours.

6. A potential implication of the findings is that OAO is attractive to use when it is applicable because:

   a. The operator can see what is happening during the optimization. With pictorial problem representation, he can make adjustments to the optimization procedure or results to compensate for limitations in problem representation more easily than he can when there is no pictorial representation.

b. The time required to train operators to use OAO with pictorial problem representation is apparently relatively short and does not require technical knowledge of the optimization algorithms.

D. RECOMMENDATIONS

If only one of the two ISC-developed aids is to be implemented on the Operational Decision Aids facility at the University of Pennsylvania, then the nonlinear programming aid should be chosen. If funds are available it would be worthwhile to implement the dynamic programming aid also so that Navy officers and R & D managers could get a feel for the way such an aid could be used. TFC decision problems that involve discrete variables should be examined for their applicability to a dynamic programming aid used in operator aided optimization mode.

ISC found that the three-type characterization of machine participation in decision making used in this report was useful. Based on this usefulness, ISC suggests that the third type, man helping machine to help man, be used in future command and control studies. ISC will assume that command and control decisions can be categorized as ISC suggests. If this assumption is valid, then ISC recommends that decisions for which the third type is appropriate be further evaluated to determine if geometric/graphic representation is the most efficient way to represent the problem in each case.

The ocean-borne enemy sensors facing a real world air strike planner are in motion during the planning and execution of the air strike. Consequently, the detection field representing the joint detection capability of enemy sensors is dynamic and not static. The problems used in the recently completed experiment show static sensors to the operators.

Representing the more realistic dynamic situation involves dynamically updating the detection contours to account for the changing positions of enemy sensors. The contour drawing algorithm for a machine stored analytic function representing joint sensor detection capability is relatively complex. In order to get the best "fit," i.e., best representation of the contours, it is currently necessary to vary a weighting factor, visually observe the contours drawn for each value of the weighting factor, and select the value that gives the best fit.

Recommended steps to be taken in implementing real time dynamics are:

1. Devise a method for automatically obtaining an acceptable fit for the contours.
2. Provide the operator with controls that will enable him to consider the sensor movement while <u>planning</u> the air strike.
3. Provide the operator with controls that will enable him to consider sensor movement <u>during</u> the air strike and issue path change directions.
4. Design an experiment that would compare operator aided optimization performance with performance of an automated algorithm.

ACKNOWLEDGEMENTS

We would like to acknowledge the support of Dr. Martin A. Tolcott, Office of Naval Research. We especially appreciate the guidance and thoughtful suggestions by Dr. Tolcott and Mr. J.R. Simpson of ONR. Their suggestions helped to improve the concepts tested and to increase the information in the analysis.

The authors must also acknowledge the contributions of Gary W. Irving, formerly of ISC. Mr. Irving was the principal developer of the concepts investigated and the models used. He also participated in the experimental design and running the experiment.

The authors take sole responsibility for the content of the report.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION

A. BACKGROUND

This is the second report by Integrated Sciences Corporation (ISC) as one of a group of contractors working on the Operational Decision Aids (ODA) program directed by the Office of Naval Research. The ODA program was initiated in 1974. It is intended to develop a variety of decision aids and test and evaluate their usefulness to the Navy. Although the program is not tied to any specific command and control hardware system, it has focused on the functions of a Task Force Commander (TFC) and his staff. The role of ISC has been to find ways to improve man-machine communication by allocating functions between man and machine that take advantage of their respective strengths.

ISC's early work on the ODA program explored the use of techniques by which a decision maker might express and communicate his perception of important relationships. ISC calls the particular techniques it has been developing "Sketch Models." A Sketch Model is essentially a "picture" that is first mentally visualized, and then drawn by a decision maker. As used here, the picture represents the decision maker's perception of the functional relationship between two or more variables, with the stipulation that the function be continuous in at least one dimension. Depending on the application, a Sketch Model can be, for instance, a single curve defining the relationship between two variables, or it can be a family of parameterized curves, or it can be a two-dimensional projection of the iso-"altitude" contours of a three dimensional function.

ISC's first investigation evaluated the ability of human operators to generate Sketch Models of bivariate Gaussian density functions from sampled data. In an experiment, a group of subjects were found capable of developing accurate Sketch Models of one type of well-behaved (i.e., unimodal and symmetric) three-dimensional function. These Sketch Models developed by the subjects from small samples of the underlying functions estimated those functions at least as well as, and in some cases better than, the statistical technique of maximum likelihood estimation.

A second study was undertaken to extend those results by investigating the ability of human operators to generate Sketch Models of less well-behaved functions, i.e., multimodal and unsymmetric. Experimental results gave a very strong indication that the subjects were able to produce accurate (as measured by percent volume error) Sketch Models for a highly irregular (multimodal and unsymmetric) function representing the joint detection capability of multiple sensors (Reference 1).

The study also sought to evaluate the usefulness of those Sketch Models for a decision task represented by an air strike path optimization problem. The experimental results did not prove or disprove the usefulness of the Sketch Model technique as an aid to the decision problem. There were two reasons. One was insufficient data, due to severe attrition in the pool of subjects trained in Sketch Modeling during a lengthy hiatus caused by a series of hardware malfunctions. The second problem surfaced as soon as the data were analyzed: it appears that the problems were too easy for the subjects. It was therefore difficult to ascertain the usefulness of Sketch Models as an aid to the strike path selection problem.

B. TYPES OF DECISION MAKING

In this study ISC suggests that machine participation in decision making can usefully be examined by distinguishing three types. These are shown in Figure 1. ISC has used this proposed classification scheme in the work reported here. In ISC's opinion these categories were appropriate for the command and control functions studied and appear to be a useful basis for most command and control analyses.

The three types distinguished here correspond to different degrees of machine participation in decision making. The unaided, or "man-only," type depicted in the figure is relied upon when automated aids either do not exist or are inappropriate to the user's needs or preferences. Applications of machine aided decision making or "machine helping man" have proliferated due to rapid advancements in computer capabilities and operations research

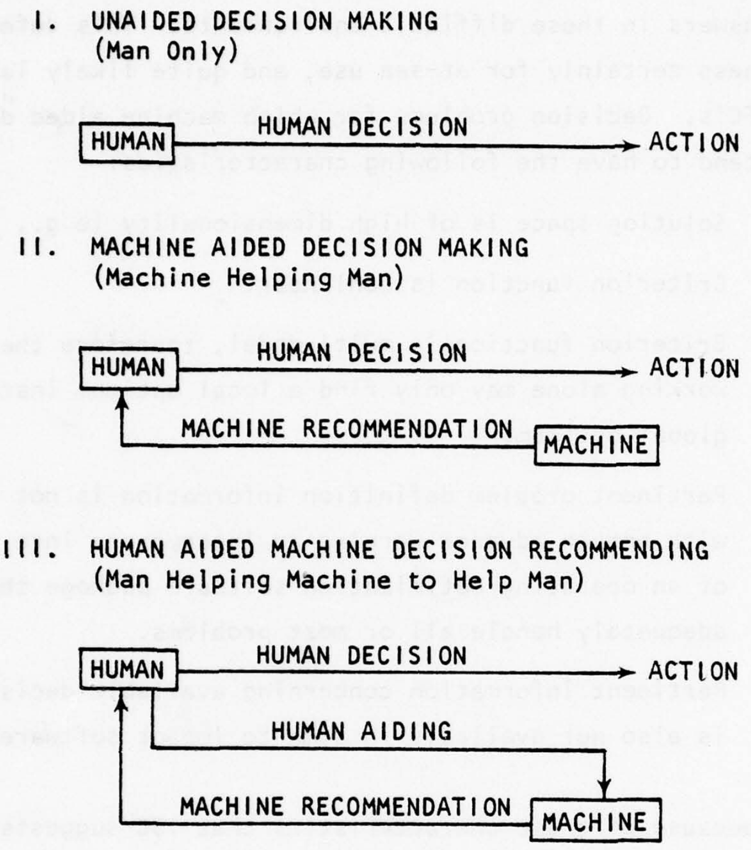


Figure 1. Types of Aided Decision Making.

techniques. Managers like to use a machine to find a problem solution when they understand what the machine does and believe in the model used by the machine. Examples of the machine helping man, as the term is used here, include the many uses of computers to solve product mix problems by linear programming methods and solution of transportation network problems by dynamic programming.

ISC believes that its characterization of decision aiding types can help to distinguish actual TFC decision environments appropriate for each type. In particular, ISC believes that there are a large number of decision environments for which the second type is not useful. This is because the methods available are often too slow, require too much hardware capability, and give questionable answers in these difficult environments. This defeats their cost-effectiveness certainly for at-sea use, and quite likely land based planning use, by TFC's. Decision problems for which machine aided decision making is inadequate tend to have the following characteristics:

1. Solution space is of high dimensionality (e.g., ≥ 5).
2. Criterion function is nonlinear.
3. Criterion function is multi-modal, therefore the machine working alone may only find a local optimum instead of the global optimum.
4. Pertinent problem definition information is not available with enough advance warning to incorporate into the design of an operating optimization software package that would adequately handle all or most problems.
5. Pertinent information concerning available decision options is also not available in time to impact software development.

It is because of these characteristics that ISC suggests and concentrates in this report on defining the third type. ISC believes this is a somewhat new and relatively powerful distinction. The "man helping machine to help man" type of decision aiding finds favor when machine aided decision making is inadequate and one or both of the following circumstances apply:

1. The decision maker does not have a good understanding of how the machine works to obtain a solution. This is often the case when an iterative optimization procedure is used.
2. The decision maker believes that the model used by the machine is not adequately representative of the real world and therefore he wishes to be able to use his knowledge of the real world to compensate for the model's limitation.

C. THE CURRENT STUDY

ISC's previous work on the ODA program had established that humans were adept at perceiving and sketching complex functional relationships when data that could be used to estimate the function were presented to the human in geometric/graphical format. The question became the following: How useful is this human capability? Therefore ISC proceeded to define (a) two decision aids that would use the human capability to solve an experimental problem that could also be solved by a fully automated algorithm, i.e., machine aided decision making and (b) an experiment that would compare decision performance with and without the aids. Several key assumptions were made:

1. Some of the complex problems requiring decisions by the Task Force Commander can be treated by analytic methods.
2. Special purpose algorithms could be constructed to solve <u>any</u> problem, but there are so many variations on problems that it would be impossible or uneconomic to have a special purpose algorithm to solve <u>every</u> problem. Therefore, it is likely that the TFC will have a general purpose algorithm available to solve each generic class of problems.
3. The Task Force Commander of the 1980's will have a general purpose computer and computer driven display at his disposal.

It is important to understand the nature and purpose of the experiments reported in this third ISC study for the ODA program. Although ISC used much of the structure and characteristics of a real-world situation, the experiment was deliberately limited and therefore, in a sense, artificial. The problem situation used in the experiment is the selection of (a) an air strike path through a field of ten enemy sensors and (b) aircraft speeds on each leg of the path. (Hereafter in this report, the selection of path and speeds is abbreviated to "selection of path.") Many aspects of real-world air strike planning were not included in the experimental problem, e.g., aircraft altitude, specific locations of enemy weapon systems and such real-world systems as electronic countermeasures. Also, the design of the experimental problems made certain perfect-information assumptions in order to simplify the analysis:

1. Sensors are stationary throughout the air strike and the location of each sensor is perfectly known.

2. A function specifying composite sensor detection performance is perfectly known.

3. Fuel consumption characteristics of the strike aircraft are known.

The utility function used to compute the goodness of each candidate strike path was composed of only two factors. It was formulated to reward low probabilities of detection along the strike path and high values of fuel remaining when the aircraft reach the target. Utility in the real world would usually be determined by more than just two factors. Expected aircraft attrition would usually be a factor in the utility function; it is not included here because the problem does not include enemy weapon systems. Therefore, the utility function is a simplified version of real world considerations.

The experiment was principally designed to contrast decision performance obtained with aids of the second and third type. A general purpose algorithm is available to solve the experimental problem in the second type or "machine helping man" mode of aided decision making. However, a "man helping machine to help man" aid which uses man's ability to visually perceive complex functional relationships is also available. This aid uses the same general purpose algorithm as is used in the "machine helping man" case. However, the man now controls the use of the algorithm instead of letting it run open-loop. The experiment designed by ISC compares decision performance for both types of aids.

These aids were not intended for immediate use in the design phase of the current Navy Tactical Flag Command Center (TFCC) project. The problem and the utility function are scaled-down versions of real-world considerations. A much larger effort would have been necessary to design an aid suitable for the TFCC project. However, the problem and utility function do have the five characteristics enumerated in Section I-B for which the "machine helping man" mode of aided decision making tends to be inadequate. Also, the "man helping machine to help man" aid designed by ISC does help the operator to understand how the machine works to obtain a solution and it does enable the operator to compensate for the model's limitations. Therefore, it was <u>not</u> the purpose of the current work to produce an aid that would be ready for TFC use upon completion of the work. Instead, the purpose of the work was to use the experimental results to draw useful inferences about the relative values of the two types of aids for the type of situation in which both are used in the experiment.

D. THE REPORT

All phases of this study are documented in the following sections. Section II describes more fully the way the basic path optimization problem was put together: it explains how the ONRODA Scenario was adapted, distinguishes at more length between the various ways of determining strike path solutions, explains the analytical models for single sensor detection performance and aircraft fuel consumption, and characterizes the utility function developed to evaluate strike paths. Section III details system operation; it comprises step-by-step explanations of how path solutions were obtained by the general purpose optimization algorithms and by subjects controlling the algorithms. Sections IV and V outline the experiment and the data analyses performed, respectively. Section VI interprets the results insofar as the data warrant. The appendices document the algorithms used to implement aspects of the study and the training material provided to operators. The algorithms are provided for the reader interested in seeing how certain operations research techniques were adapted.

II. CONTEXT FOR THE EXPERIMENT

A tactical decision task was defined to investigate the usefulness of decision aids that make use of man's ability to visually perceive complex functional relationships. The task was that of optimizing an air strike path through a defender's multi-sensor detection field. This section describes the task scenario, the system concepts that represent different ways of optimizing a strike path, and the models of the scenario variables that constitute the experimental vehicle. Each model described reflects certain assumptions made about the behavior of the scenario variable. These assumptions, in turn, were adopted to keep the test vehicle simple, rather than to faithfully model the variables' "real world" performance. The utility criterion function, ultimately used as a performance measure, is also described here in terms of its supporting models.

A. AIR STRIKE SCENARIO

The problem selected, implicit in the ONRODA scenario, was that of optimizing an air strike path between a strike launch point and a target. The evaluation of the path depended on the probability of an aircraft's being detected by the enemy and the amount of fuel consumed by the aircraft along the strike path. Accordingly, certain elements of interest, particularly the scenario geography, were extracted from the ONRODA Warfare Scenario (Reference 2), and other details, described below, were added. The scenario developed here assumes that the decision has been made to conduct an air strike against ONRODA, so that investigating the relative usefulness of competing decision aids in this study means applying them to one aspect of the operational implementation of the decision to strike.

Figure 2 shows the 500-by-500 n.m. portion of the ONRODA warfare scenario area map used to provide the geographical context for this study. The boundaries provide an area west of ONRODA for the selection of strike launch points and (it is assumed) enough room to plan strike paths that do not violate the ORANGE sanctuary.

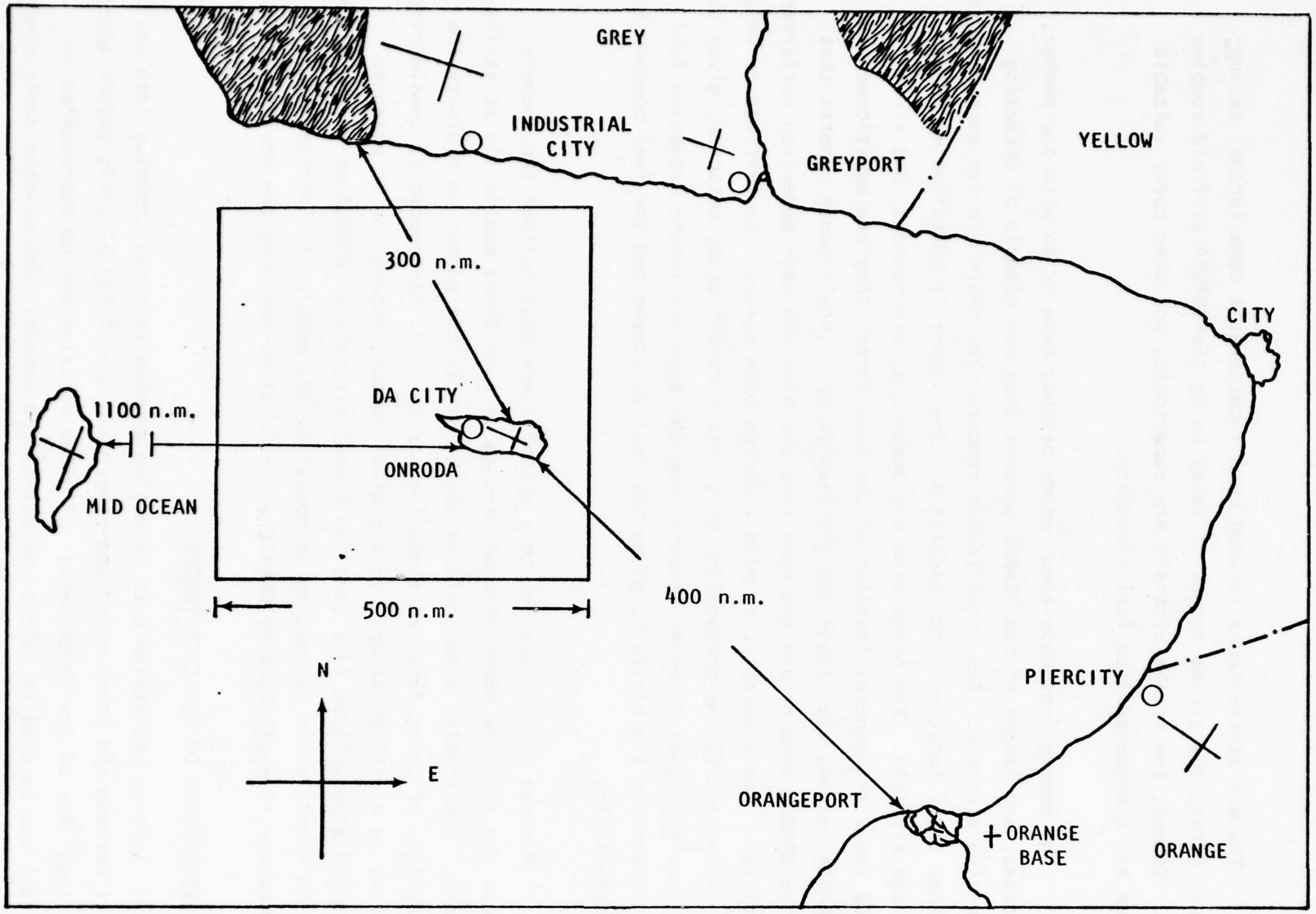


Figure 2. Strike Scenario Geography.

The air strike scenario used here incorporates some further assumptions. First, the strike target is taken to be the ONRODA airfield complex only. Second, the strike aircraft are supersonic, and they carry suitable stores and a predetermined fuel allotment.

The assumptions made about enemy defense have to do with the number, locations, and ranges of the ORANGE sensors that are capable of detecting the strike aircraft. Own intelligence reports that there are ten such sensors and that their locations are pinpointed. One sensor is installed on ONRODA near the airfield. The other nine are ocean-platform mounted, and since ORANGE knows the general location of the task force, they are positioned west of ONRODA between the island and the task force. Intelligence reports that all the ORANGE sensors are the same type and have the same detection performance capability. The problem is to plan a strike path against the airfield on ONRODA that (a) minimizes the probability of strike aircraft being detected, given the locations and types of enemy sensors, and (b) does not impose excessive fuel reuqirements on the aircraft, given the fuel allotment and the fuel consumption characteristics.

Further assumptions for this scenario are that neither the enemy's defense nor airborne enemy aircraft are to be considered explicitly as strike factors. Implicitly, enemy defense capability is one reason to minimize the probability of being detected along the strike path, tantamount to considering surprise as a strike factor. In a similar manner, attempting to postpone detection also affords less time for ORANGE aircraft on ONRODA or on the mainland to react, while attempting to conserve fuel enables the strike aircraft to maneuver if challenged by ORANGE aircraft after reaching the strike target.

B. DESCRIPTION OF CONCEPTS TESTED

Solving the strike path selection problem requires choosing path way points between the start point and target and specifying aircraft speeds along each leg. Any of the three types of aided decision making represented in Figure 1 can be used to select way points and speeds. Abbreviated descriptors used in this report that are equivalent to the three types shown in Figure 1 are given below.

Types of Aided Decision Making (from Figure 1)	Equivalent Descriptors Used in this Report
Man Only	Operator Unaided (OU)
Machine Helping Man	Automated (A)
Man Helping Machine to Help Man	Operator Aided Optimization (OAO)

Two types of general purpose optimization techniques that can be used to implement the automated aid are nonlinear programming and dynamic programming. The structure of the path selection concepts tested is organized around the two general purpose optimization algorithms used to solve the problem, namely, a nonlinear programming (NP) algorithm and a dynamic programming (DP) algorithm. A "tree" representation of the structure is shown in Figure 3.

The NP algorithm requires two inputs. One is specification of the "starting point," namely the positions of the way points and the speeds along each leg. The other is a convergence criterion. The algorithm begins to iteratively optimize the path to find the highest path utility as soon as it obtains a starting "point." It stops when the convergence criterion is met. Then the algorithm obtains a new starting point and begins to optimize again. In the automated mode the algorithm contains a procedure for selecting the starting point and a convergence criterion. ISC obtained test data on two procedures for selecting the starting point and three convergence criteria. In operator aided optimization mode, the operator selects the starting point and decides when to stop the optimizing done by the algorithm. (The same algorithm is used for the automated mode and the OAO.) Then he selects a new starting point and starts operation of the algorithm again. In operator unaided mode, the operator specifies the path way points and speeds on path legs which constitute his estimate of the best solution. This is a one-step process; there is no iterative optimization as occurs in the automated mode and OAO. The operator unaided mode corresponds to the procedure that would be used today by a Task Force Commander.

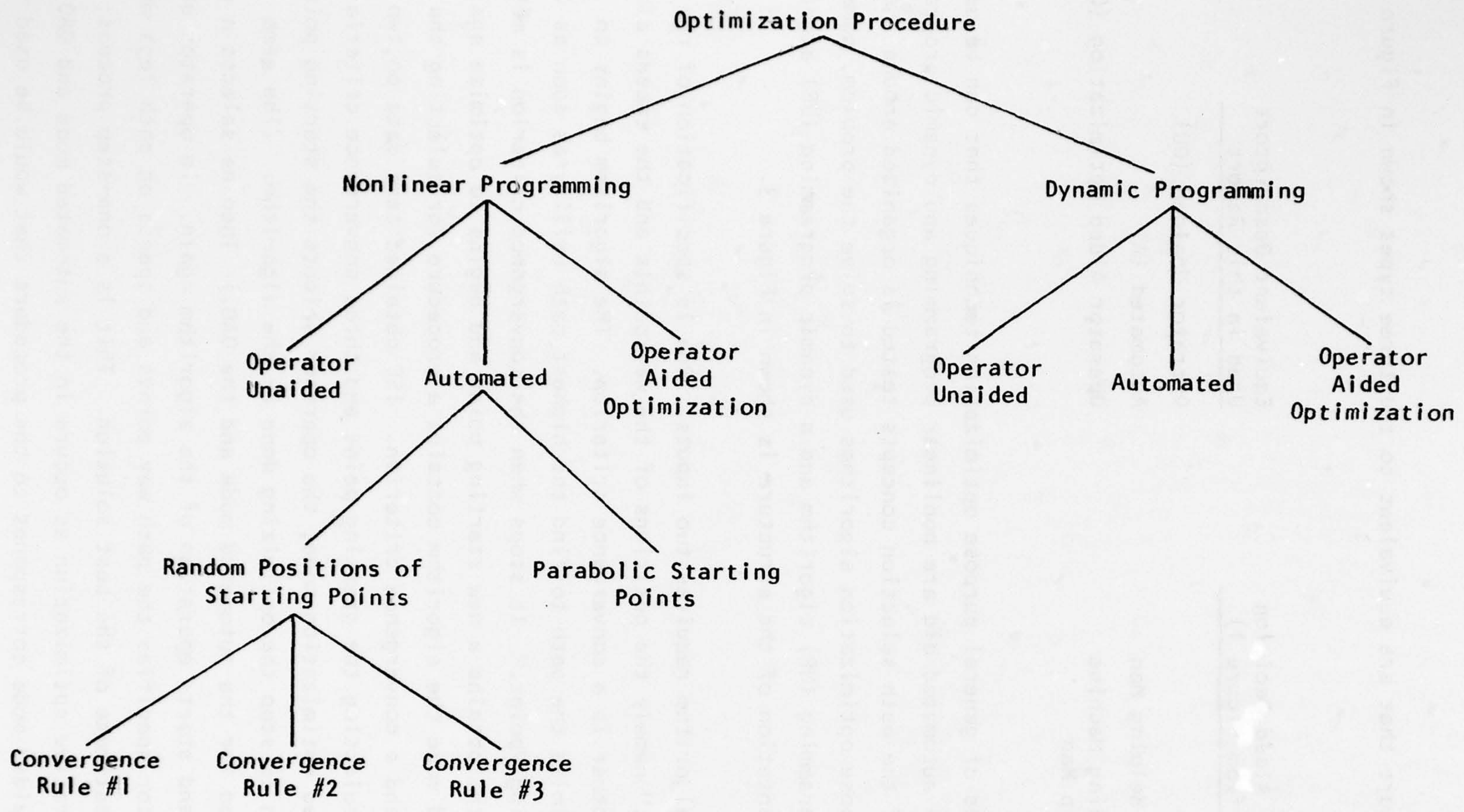


Figure 3. Structure of Optimization Concepts Tested.

The DP algorithm finds for a given set of constraints the best path composed of (a) legs from point to point on a specified grid and (b) the best of a set of three specified speeds for each path segment between two adjacent grid points. In the automated mode the algorithm considers all the points on the grid for a specified spacing of grid points and all the available speeds, i.e., three speeds, for each path segment. It finds the best solution for the specified grid spacing and stops. It does not iterate as does the NP algorithm because the DP algorithm, unlike the NP algorithm, finds a global optimum for the specified grid size.

In OAO mode the operator draws a boundary around the region where he wants the DP algorithm to optimize. He then selects a grid spacing and constrains the algorithm's search by eliminating from its consideration the speeds he believes will not be in the final solution. The operator starts the algorithm and the algorithm proceeds to find the best solution for the grid spacing and constraints set by the operator. The operator views the solution and then decides on a new combination of grid spacing and constraints for (a) the bounded area to be searched and (b) speeds to be eliminated from consideration. In this fashion the operator iteratively refines solutions found by the algorithm.

In operator unaided mode the operator draws the estimated best path from point to point on a specified grid size and he specifies which of the three available speeds is to be used for each path segment between adjacent grid points. Operator unaided mode <u>resembles</u> what would be done by the Task Force Commander today. It is not exactly the same because the operator is constrained to use the specified grid points and he must select one of three specified speeds for each path segment.

C. SIMULATION MODELS AND ALGORITHMS

The "goodness" of a path generated under any of the system concepts depends on two factors: fuel consumed along the path and the cumulative probability of being detected. In order to compute a numerical value (or utility) that reflects a given path's "goodness," it is first necessary to have some way of quantifying those two factors. This was provided by a set of simulation models and computational algorithms developed for the study. Fuel consumption was modeled as a single functional relationship. The cumulative probability of being detected, however, is more complex and depends on how the characteristics of the detection field are defined. In general, this involves first defining <u>single</u> sensor performance, then defining the way a number of these single sensors combine to create a composite detection field.

The set of models and algorithms used in the study includes:

1. Single-Sensor Detection Rate Model
2. Cumulative Probability of Being Detected Algorithm
3. Fuel Consumption Model
4. Utility Criterion Function
5. Dynamic Programming Algorithm
6. Nonlinear Programming Algorithm
7. True Detection Rate Contour-Drawing Algorithm

Numbers 1 - 4 are described in this section; numbers 5 - 7 are documented in the appendices.

1. <u>Overview</u>

Figure 4 shows how the models and algorithms are used for operator aided optimization. Scenario elements (composite detection capability of the ten enemy sensors, strike launch point, and target) defining the problem are stored in the computer and are shown to the operator by means of the display (①). He enters his inputs to the NP or DP algorithm by means of the display peripherals (②). Inputs to both algorithms are the problem definition and the operator inputs.

The NP algorithm considers a candidate path, finds the cumulative probability that the air strike will be detected if that path is used, the fuel consumed on the path, and the utility of the path considering both cumulative detection probability and fuel consumption. Each path considered by the NP algorithm and its utility is displayed to the operator (③). The NP algorithm continues to find better paths, and these are continually displayed until the

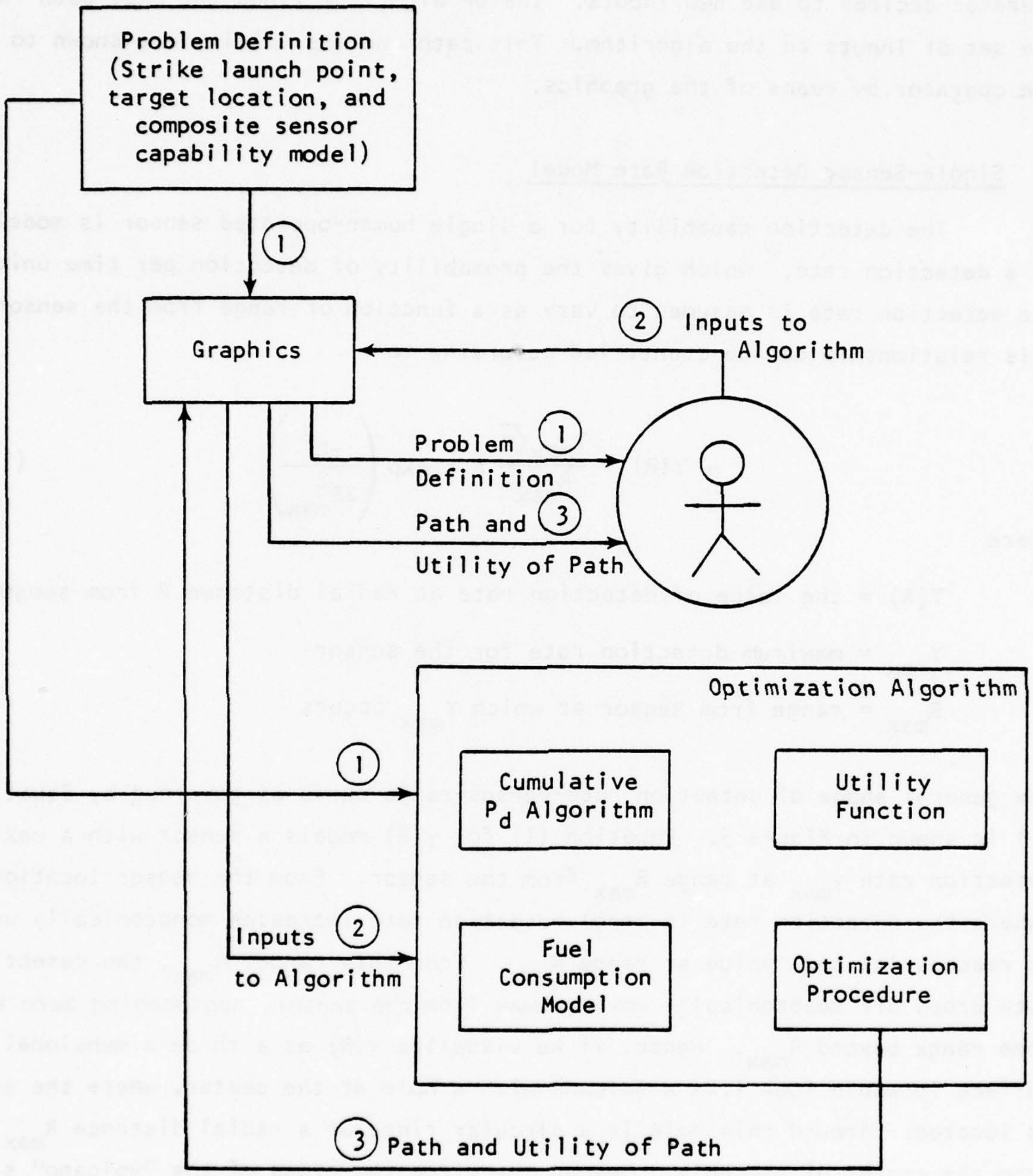

Problem Definition (Strike launch point, target location, and composite sensor capability model)
1
Graphics
2 Inputs to Algorithm
Problem Definition 1
Path and 3 Utility of Path
Optimization Algorithm
1
Cumulative P_d Algorithm
Utility Function
Inputs 2 to Algorithm
Fuel Consumption Model
Optimization Procedure
3 Path and Utility of Path

Figure 4. Interrelation of Models and Algorithms for Operator Aided Optimization.

operator decides to use new inputs. The DP algorithm finds the best path for the set of inputs to the algorithm. This path and its utility are shown to the operator by means of the graphics.

2. Single-Sensor Detection Rate Model

The detection capability for a single human-operated sensor is modeled as a detection rate,* which gives the probability of detection per time unit. The detection rate is assumed to vary as a function of range from the sensor. This relationship can be quantified according to:

$$\gamma(R) = \frac{\gamma_{max}\sqrt{e}}{R_{max}} R \cdot \exp\left(\frac{-R^2}{2R_{max}^2}\right) \tag{1}$$

where

$\gamma(R)$ = the value of detection rate at radial distance R from sensor

γ_{max} = maximum detection rate for the sensor

R_{max} = range from sensor at which γ_{max} occurs

The general shape of detection rate-versus-range curve as governed by Equation (1) is shown in Figure 5. Equation (1) for $\gamma(R)$ models a sensor with a maximum detection rate γ_{max} at range R_{max} from the sensor. From the sensor location (where the detection rate is zero) detection rate increases monotonically until it reaches its peak value at range R_{max}. From this range, R_{max}, the detection rate drops off monotonically moving away from the sensor, approaching zero at some range beyond R_{max}. Hence, if we visualize $\gamma(R)$ as a three-dimensional surface it would look like a volcano with a hole at the center, where the sensor is located. Around this hole is a circular ridge at a radial distance R_{max} from the center of the hole. Beyond the ridge the sides of the "volcano" slope downwards until "ground level" is reached.

*Detection rate is a quantitative measure of sensor performance (Ref.6) defined over the space surrounding a sensor. An intuitive understanding of detection rate, $\gamma(x,y)$, may be had by considering that $\gamma\Delta t$ is the conditional probability that a target is detected at or near (x,y) given that 1) Δt is small and 2) no detection occurred before Δt.

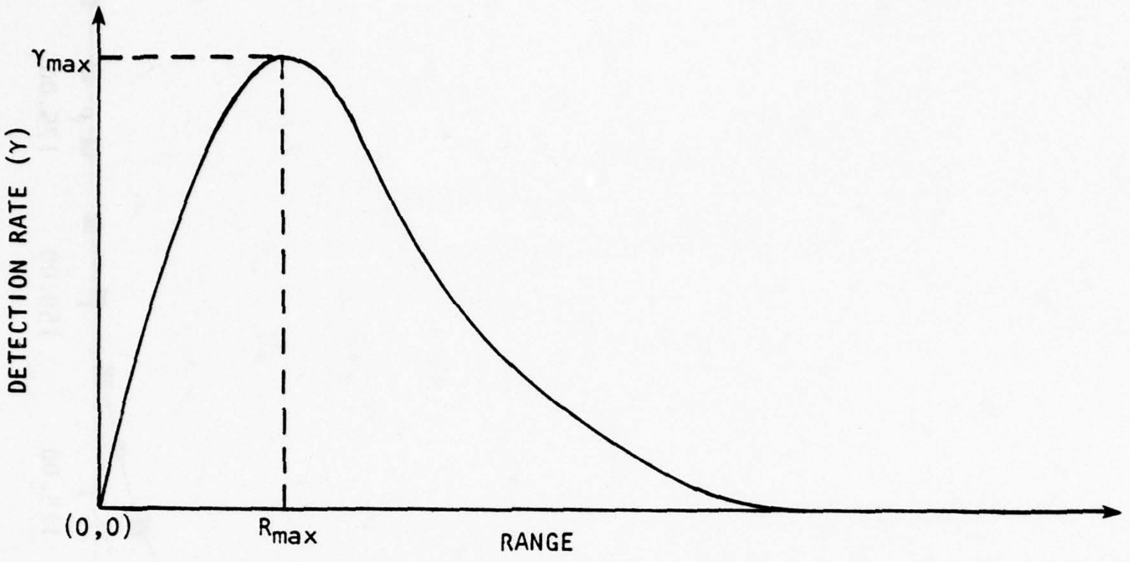


Figure 5. Single Sensor Detection Rate as a Function of Range.

For the experiment one type of sensor was defined, corresponding to R_{max} = 37.5 nautical miles. This value of R_{max} was selected for its suitability to the study. It was not intended to be the performance value for any "real world" sensor. The maximum detection rate γ_{max} was 0.1. The performance curve for the sensor type is shown in Figure 6.

Recall that the scenario specified ten enemy sensors deployed, so that if two (or more) detection ranges overlap, we are really concerned about our strike aircraft being detected by at least one sensor rather than being detected by more than one. In other words, we are concerned about the total detection rate at any point that any given set of sensor locations will produce. This composite detection rate is easily computed. The composite detection rate γ_c at a point (x,y) is the sum of detection rates at (x,y) due to each sensor. Hence, if $\gamma_i(x,y)$ is the detection rate due to sensor i, the composite detection rate at (x,y) is

$$\gamma_c(x,y) = \sum_i \gamma_i(x,y) \qquad (2)$$

Each term $\gamma_i(x,y)$ on the right hand side of Eq. (2) is obtained by transforming the radial coordinates of Eq. (1) into rectangular coordinates.

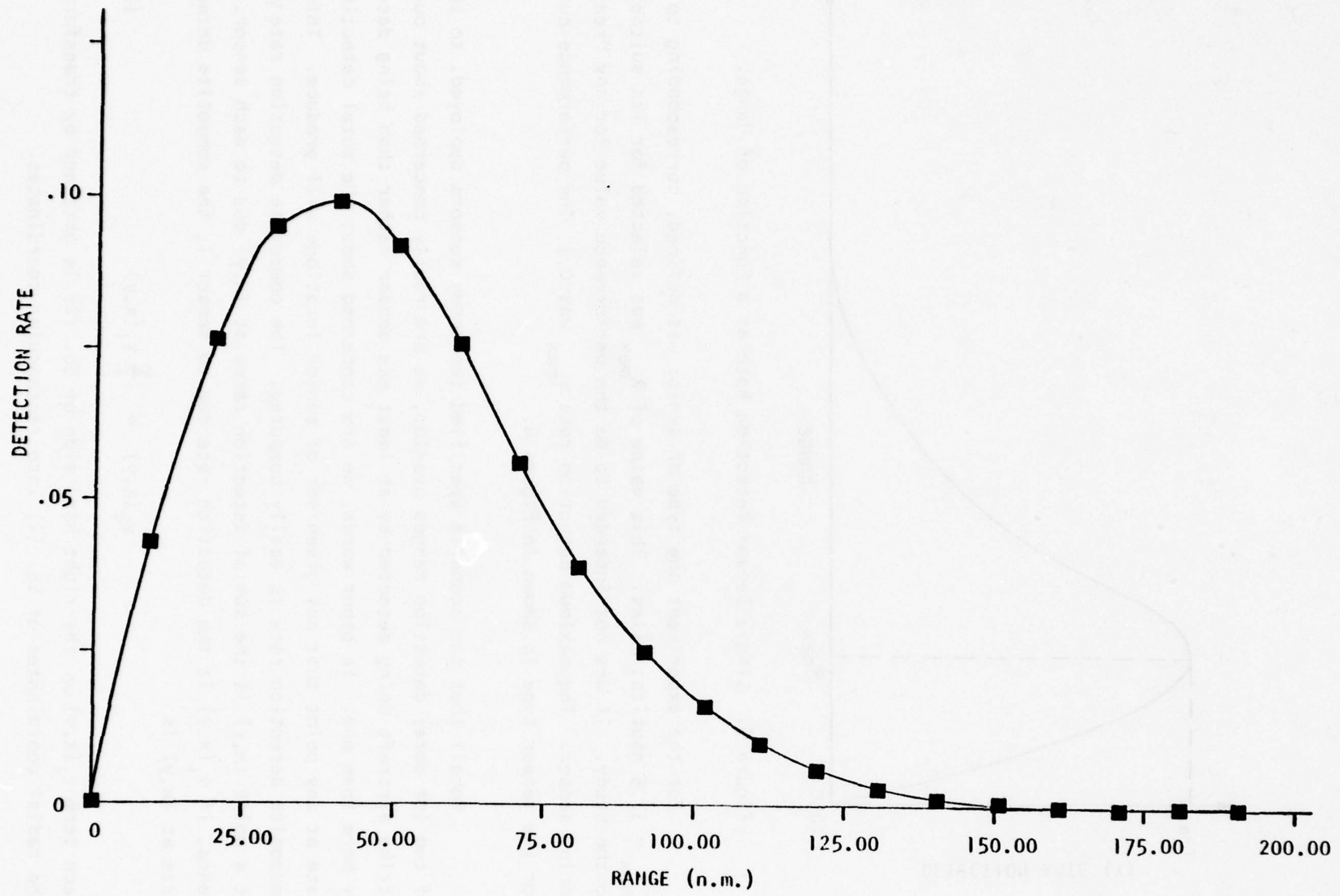


Figure 6. Performance Curve of Modeled Sensor.

The reader may question the validity of the additive operation in Eq. (2), since probabilities are not additive in general. After all, detection rate as we have defined it is the probability of detection per unit time. The justification of the operation in Eq. (2) lies in the fact that we choose Δt (see footnote on page 16) small enough such that within Δt the probability of detection by two or more sensors is negligible,* all the higher order terms in the exact expansion for the left hand side of Eq. (2) drop out, leaving the right hand side of Eq. (2).

3. Cumulative Probability of Being Detected Algorithm

For an aircraft flying an air strike path through the enemy's multi-sensor detection field, it is necessary to calculate the cumulative probability that the aircraft will be detected by the time it reaches the target. The cumulative probability that an aircraft will <u>not</u> be detected on a given leg by a single sensor is the building block used to calculate cumulative detection probability. This is:

$$P_{nd}\text{ (cumulative, no detection, single sensor)} = \exp\left[-\int_{t_0}^{t_1} \gamma[R(t)]\,dt\right] \qquad (3)$$

where:

t_0 = time at beginning of leg

t_1 = time at end of leg

For multiple sensors, the cumulative probability that an aircraft will <u>not</u> be detected on a given leg is:

$$P_{nd}\text{ (cumulative, no detection on leg)} = \exp\left[-\sum_{s=1}^{s=S}\int_{t_0}^{t_1} \gamma[R_{s_j}(t)]dt\right] \qquad (4)$$

where

S = total number of enemy sensors

*This may remind the reader of similar practices in various branches of operations research, such as queueing theory.

The cumulative detection probability for the entire path is calculated by:

$$P_d \text{ (cumulative detection on path)} = 1 - \exp\left[-\sum_{\ell=1}^{\ell=L} \sum_{s=1}^{s=S} \int_{t_{0,\ell}}^{t_{1,\ell}} \gamma[R_{s_j}(t)]\, dt\right] \quad (5)$$

where:

L = number of legs in path

4. Fuel Consumption Model

The rate of fuel consumption was calculated in accordance with Equation 6 below:

$$\text{Fuel rate} = 0.0377\, v^2 - 16.57v + 3869 \text{ (lbs/hr)} \quad (6)$$

where,

v = aircraft speed in knots

Fuel used per path leg is:

$$\text{Fuel consumed } (\text{leg}_i) = \frac{\left(\text{Leg Length}\right)\left(\text{Fuel Rate } (v_j)\right)}{v_j} \quad (7)$$

Operators using the NP aid were allowed to select any speed from 250 to 1,000 knots for each leg. Operators using the DP aid were allowed to select high, medium, or low speed for each leg. These speeds were 1000, 625, and 250 knots respectively. The fuel consumption rates for these speeds are listed in Table 1. The second and third columns of Table 1 are equivalent; they are simply expressed in different units for easier reference.

The amount of fuel that an aircraft carries on each mission is proportional to the range from the strike launch point to the target. Thus, if the range is doubled, the fuel allowance for the mission also doubles. The fuel allowance for each nautical mile between the air strike start position and

Table 1. Fuel Consumption Rates

Velocity (knots)	Fuel Consumption Rates	
	lbs/sec	lbs/n.m.
250	0.5785	8.331
625	2.289	13.183
1,000	6.944	24.999

target was 39.69 pounds. This permits the aircraft to do some high-speed manuevering, but sustained high-speed travel is discouraged by the fact that allotted fuel would run out before the aircraft could accomplish the mission or return to the carrier.

5. Utility Criterion Function

A utility criterion function with which to measure the performance under each of the system concepts in the experiment was defined. The problem was to select an optimal air strike path, so an appropriate utility criterion function is one which measures the "goodness" of such an air strike path. The two variables selected to determine the goodness of an air strike path were fuel consumption along the path and probability of being detected by the enemy sensors (Sections 3 and 4, preceding). Since the utility function was pre-specified to measure the goodness of any proposed path, no inputs were elicited from operators as to desirable values of the two variables. The following definition of the utility criterion function, U, incorporates a tradeoff between minimizing the probability of being detected by enemy sensors on one hand and maximizing the fuel remaining upon arrival at the target on the other.

$$U(F,P) = \left[\frac{(a-b)D - F}{2(a-2b)D}\right]^{(.01 + 4.95P)}, \text{ if } (a-b)D - F \geq 0 \qquad (8)$$

$$= 0, \text{ if } (a-b)D - F < 0 \text{ and DP aid is used}$$

$$= (a-2b)D - F, \text{ if } (a-b)D - F < 0 \text{ and NP aid is used}$$

where

> F = total amount of fuel consumed upon arrival at target
>
> P = cumulative probability of being detected by enemy sensors
>
> D = distance between strike launch point and target
>
> a = fuel allowance/n.m.
>
> b = fuel consumption/n.m. at an achievable speed resulting in the lowest fuel consumption per unit distance traveled

For each mission the fuel allowance is proportional to the shortest distance between the air strike launch point and the tarnet $(a \cdot D)$. The absolute minimum fuel that has to be preserved in order to return from the target is $(b \circ D)$. Hence, $(a - b)D$ is the maximum amount of fuel available for maneuvering to the target, and $(a - 2b)D$ is the maximum amount of fuel remaining upon return to the carrier. Note that if the aircraft runs out of fuel before returning to the carrier, the resulting utility is:

1. Zero for the DP algorithm
2. Negative and equal to the difference between minimum possible fuel usage and actual usage for the NP algorithm. This is a device to increase convergence speed.

For the experiment, $a = 29.7$ lbs/n.m., and $b = 8.3$ lbs/n.m., corresponding to a velocity of 250 n.m./hr.

The utility function takes on any value between 0 and 1, with higher utility values corresponding to "better" paths. As the probability of being detected by enemy sensors decreases, the utility value goes up. Also, if the probability remains constant, the utility value increases as fuel consumption drops. It is obvious why it is desirable to minimize the probability of being detected by enemy sensors. The rationale for encouraging fuel preservation is that if detection occurs at any time up to arrival at the target, there should be as much fuel left as possible in order to do some flight maneuvering to try to return safely.

In general the two goals of minimizing fuel consumption and minimizing the probability of being detected are incompatible. A nontrivial optimal air strike path thus requires a reasonable compromise between the two goals. The utility function was designed as representative of the class of functions useful in designing an air strike path through a multi-sensor field, and Eq. (8) embodies a trade-off between remaining fuel and cumulative probability of being detected.

III. SYSTEM OPERATION

Important characteristics of NP and DP methods for solving the scenario problems are discussed in this section. Documentation of the step-by-step operation is given for the OAO concepts that use the DP and NP algorithms. The emphasis in the documentation is on the display concepts used. This section also discusses the relevant considerations for the choice among alternative input rules for NP automated concepts.

A. NONLINEAR PROGRAMMING OPTIMIZATION

1. Characteristics of the Nonlinear Programming Optimization Technique Used in the Experiment

The setup for the nonlinear programming technique includes the starting "point" or first trial solution. In the air strike problem, the starting "point" is (a) five path legs connecting the air strike launch point and the target and (b) speeds for each leg. The legs are specified by picking four "way points" between the launch point and target. Speeds are selected from a range of 250 to 1000 knots. After the start point has been specified, the NP technique operates to find a better combination of way points and speeds. It does this by exploring changes in the location of each way point and the speed for each leg. Each exploration involves a single combination of way point and speed changes. Therefore, improvement in the air strike path takes place slowly over many explorations, i.e., trials. An advantage of NP is that it considers all the points in the geographical region it explores instead of just a set of grid points and all speeds within the aircraft's capability instead of just a few. A disadvantage is that the "solution" will be best for the region explored but that better solutions may exist in unexplored regions that the algorithm cannot "see" to explore and the NP technique is often unable to direct itself to look in these unexplored regions. In optimization jargon, NP may find a local optimum but not the global optimum.

2. Automated Optimization Using the NP Algorithm

The approach used in selecting the NP algorithm was the following: "What type of technique would you want if you knew nothing about any given specific problem to be solved and you wanted the technique to be relatively equally adept at solving a broad diversity of problems?" This approach is consistent with the assumptions in Section I-C made about the types of TFC problems for which optimization aids would be appropriate and the type of algorithms that would be available to the TFC. Answering this question can be broken down into the five component parts of a nonlinear programming algorithm: Selecting a starting point, selecting a search direction, selecting a step size, stopping criterion function, and selecting a new starting point for the next iteration. Table 2 expresses the approach for each component. Components 1, 2, and 3 are

Table 2. Algorithm Design Guidance for the Components of the NP Algorithm.

	COMPONENT	ALGORITHM DESIGN GUIDANCE
1	Select a starting point	Since no problem specific information is available other than the scales and upper or lower bounds of the dimensions of decision space, a random starting point selection procedure is required.
2	Select a search direction	Since it can not be assumed that either first or second derivatives will be continuous over the domain of the criterion function (e.g., Sketch Models), a gradient-free approach is required.
3	Select a step size	Since it can not be assumed that first or second derivatives will be continuous over the domain of the criterion function, a gradient-free approach is required. (Same as for component (2) above.)
4	Convergence stopping rule for automated optimization	A percentage (e.g., $\leq$ 5%) of change in either the length of the search vector or the value of the criterion function is allowable; also specifying an amount of time for each iteration regardless of progress is allowable.
5	Selecting a new starting point for next iteration during automated mode	Only information available is criterion function values achieved with other starting points; otherwise same as (1) above.

used in operator aided optimization (OAO) and automated optimization. Components 4 and 5 are specified only for automated optimization since the operator performs these functions during OAO.

A baseline approach for automated optimization was designed to be completely compatible with the assumption of zero problem-specific information. It consists of the following algorithms for the nonlinear programming components:

1. Selecting a starting point - A starting point was selected from a uniform random density function defined between the lower and upper bounds of each of the dimensions of the solution space.

2. Selecting a search direction - Rosenbrock's method of orthogonal directions was selected for this component (Reference 3). This procedure is quite competitive among gradient-free approaches working on high dimensional problems. (See Reference 4.)

3. Selecting a step size - Rosenbrock's method, experimented with over a variety of problems, recommends increasing step size by a multiple of 3 when successful and decreasing by 0.5 when unsuccessful.

4. Convergence criterion - The search was determined to have converged when the length of the search vector had not changed by more than 1% in five successive criterion function evaluations.

5. Selecting a new starting point - The time assumed reasonable for providing an answer (e.g., 15 minutes) did not allow a substantial number of iterations (e.g., ≥ 10). Thus it was not feasible to use information concerning the success of previous starting points in the selection of the n^{th} starting point, and each successive starting point was selected from the same uniform random density function described in (1).

The baseline approach was given the name "Random Starting Point."

In order to evaluate the impact of some problem-specific information on the performance of the automated concept (and therefore on the results of the OAO experiment) the following modifications to the baseline approach were evaluated:

1. Changes in the method for selecting a starting point: The Random Starting Point rule produced nonsensical air strike paths which cross over themselves (perfectly valid under the zero problem-specific information philosophy). This was a controversial component and deserving of additional investigation. Therefore the starting points were also selected by a pseudo-random process by which the path of the air strike was made to conform to straight line segments connecting points on a parabola that passed through the launch and target points. This eliminates the crossing-over aspects of the path. The velocities of the starting point were selected by a linear pseudo-random procedure which minimizes the variance of the starting velocities. The name given to this concept was "Parabolic Starting Point."

2. Changes in the algorithm for convergence criteria: This was deemed the component second-most subject to controversy. The reason was that the automated concept could be considered at an unfair disadvantage with respect to OAO if the selected convergence criterion produced poorer performance than some other criterion. Therefore two other values of convergence criteria (.1% and 5%) were evaluated. These in combination with the 1% used in the baseline approach covered a range of values and allowed the estimation of a convergence criterion value that maximizes performance of the automated concept.

The several versions of the automated concept were tested in the following sequence:

1. Performance on five trials for each of twelve problems was obtained using the Random Starting Point rule for each of the three convergence criteria, namely, 0.1%, 1%, and 5%. The result of these trials was that the 1% convergence criterion was superior to the other criteria and therefore the data from the runs with this criterion was used to represent automated NP performance for the Random Starting Point rule.

2. The 1% convergence criterion was used on five trials for each of the same twelve problems for the Parabolic Starting Point rule.

3. Operator Aided Optimization Using the Nonlinear Programming Algorithm

At the beginning of a problem the display appears as shown in Figure 7. The path from launch point to target is a straight line with way points indicated at 1/5, 2/5, 3/5, and 4/5 of the straight line distance. Speed for each leg is initially set by the program at 600 knots as indicated on the plot at the left in Figure 7. The operator uses the appropriate buttons on the function button box (see Figure 8) and the joystick to change the position of the four way points. He uses the appropriate function buttons and the keyboard to change speed on any leg.

The operator's purpose is to direct the NP technique to investigate as many reasonable potential solution regions as possible in 15 minutes, which is the length of a trial. As soon as the problem is shown on the display, the operator must decide what region he wants to explore first. His training instructions are to pick the region that he thinks is most likely to contain the best solution. He then changes the locations of the way points and speeds prior to starting the NP algorithm. The resultant path and speeds constitute his estimate of the best solution and correspond to the Operator Unaided concept.

At the beginning of the problem the three buttons designated as EVALUATE/HALT, CHANGE VELOCITY and MOVE WAY POINT are lighted on the box. In order to move a way point, the operator pushes that button. When this is done the four buttons marked 1, 2, 3, and 4 become lighted. He then pushes the button corresponding to the point to be moved, i.e., 1, 2, 3, or 4. Way point 1 is the one closest to the beginning of the strike path and 4 is nearest the end (ONRODA Island). Moving the way point is accomplished with the joy stick. When a single way point is changed a second way point can be changed by pressing MOVE WAY POINT and the appropriate number of the way point. The act of pressing MOVE WAY POINT records the position of the last way point that was changed.

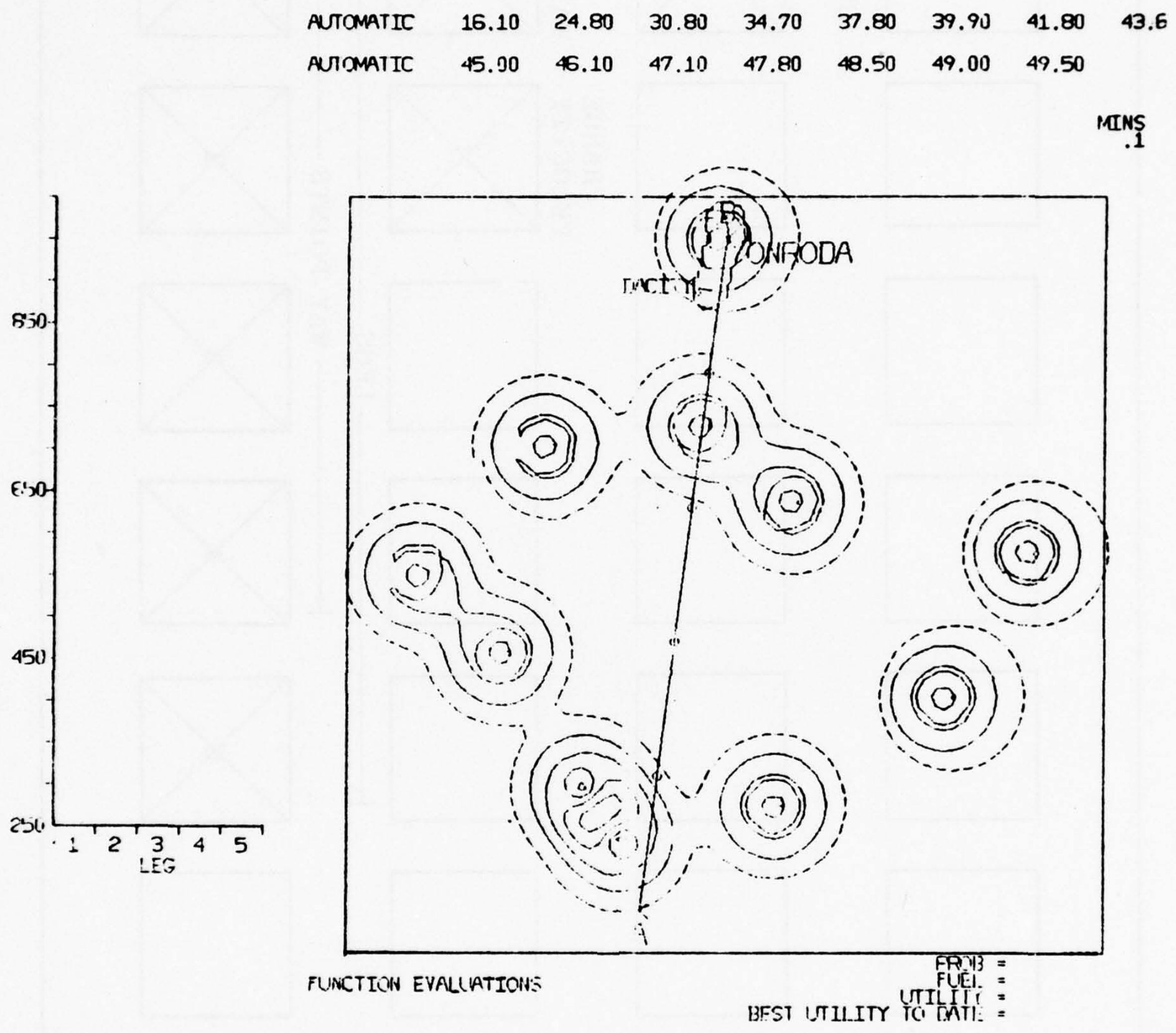

AUTOMATIC 16.10 24.80 30.80 34.70 37.80 39.90 41.80 43.6
AUTOMATIC 45.00 46.10 47.10 47.80 48.50 49.00 49.50
MINS
.1
ONRODA
850
650
450
250
1 2 3 4 5
LEG
FUNCTION EVALUATIONS
PROB =
FUEL =
UTILITY =
BEST UTILITY TO DATE =

Figure 7. Display Appearance at Beginning of Problem to be Solved Using NP Algorithm.

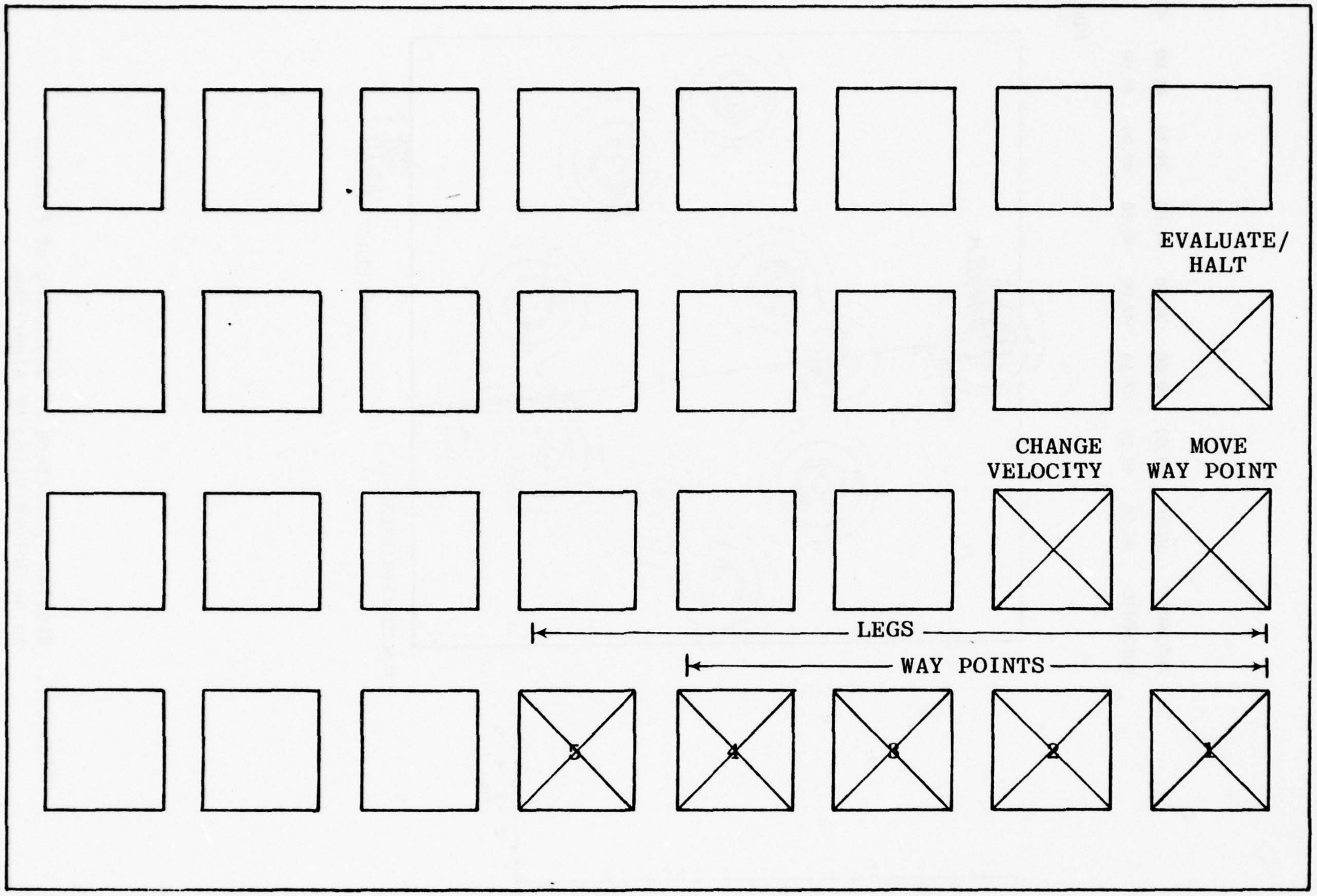


Figure 8. Function Button Configuration for the Nonlinear Programming Optimization

It often happens when using the aid that the angle between adjacent legs is 90° or less. A real aircraft cannot execute the turns indicated by the connection of the straight-line path segments in such cases. It would have been possible to improve the realism of the aid by having the computer automatically draw an arc corresponding to an aircraft's turning capability to connect the path segments. This was not done because the lack of realism did not interfere with the purpose of the work, namely, the comparison of the different kinds of decision aids.

To change a speed on one of the five legs, the operator pushes CHANGE VELOCITY. The five buttons marked 1, 2, 3, 4, and 5 then light. Leg 1 refers to the leg closest to the path beginning point and leg 5 refers to the path closest to the end point (ONRODA Island). He then pushes the button corresponding to the leg for which he wants to change speed and:

1. Uses the teletype keyboard to input the speed he wants used on the selected leg. A decimal point is put at the end of the number. (This is essential!)
2. Pushes the teletype key marked "CR."

Thus, if he wanted to change the speed on leg 3 to 850 knots, he would:

1. Press function button CHANGE VELOCITY
2. Press function button "3"
3. Press teletype key "8"
4. Press teletype key "5"
5. Press teletype key "0"
6. Press teletype key "."
7. Press teletype key "CR"

When the operator has changed all the way points and speeds to those he wants, he then presses the function button marked EVALUATE/HALT. The NP algorithm will begin to operate, i.e., EVALUATE using the starting point consisting of the four way points and five speeds. Once the algorithm has begun operating, only the EVALUATE/HALT button will remain lighted and the only control at the operator's disposal is to halt operation by pushing this button.

The primary indicators that the operator uses to decide to halt the algorithm are the displays of the number of function evaluations and the utility of the latest trial solution. In general, a plot of utility versus function evaluations would appear as shown in Figure 9. The subject should stop the algorithm when it reaches the point shown in Figure 9 because there will be little more utility to be gained by letting the algorithm continue. He should then input a new set of way points and speeds and start the algorithm again.

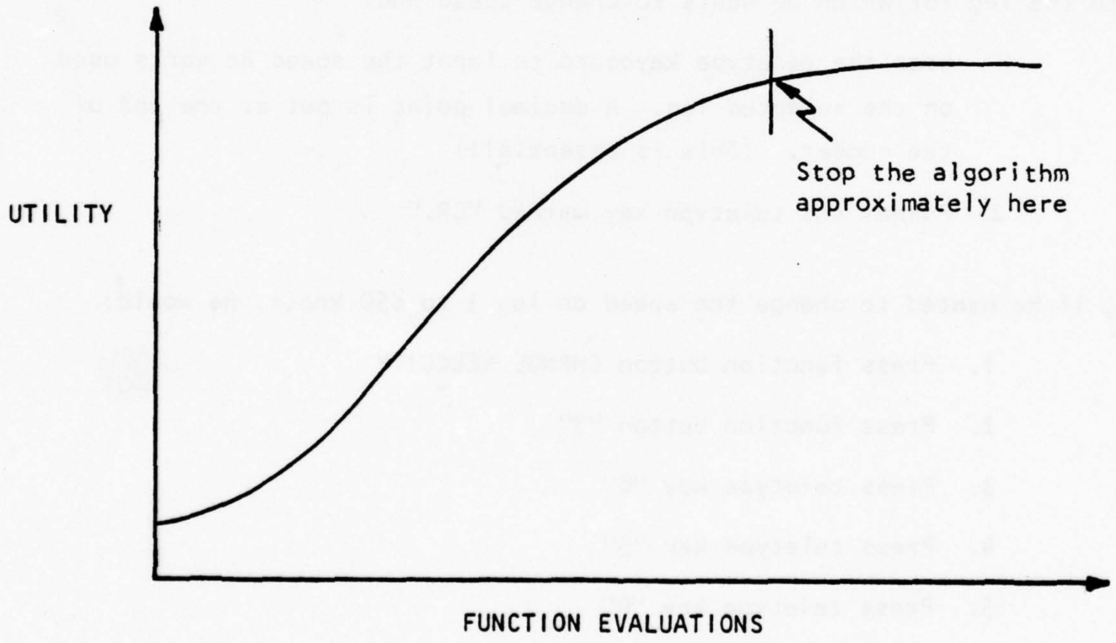


Figure 9. Typical Plot of Utility Versus Function Evaluation.

As the algorithm operates, he can see variable length arrows appearing briefly at each way point. These represent potential changes in the location of a way point being considered by the algorithm. When utility levels off, the magnitude of changes in the following will also become small:

1. Value of "Prob," i.e., the probability that the air strike will be detected prior to arrival at the target.
2. Value of "Fuel," i.e., the fuel that will be consumed for the latest trial solution.
3. Speed changes indicated on the speed/leg graph.
4. Lengths of arrows appearing at each way point.

While the algorithm is operating on the first set of way points and speeds input by the operator, he should count the number of regions that could reasonably be expected to contain the best path. Dividing 15 minutes by the number of regions to be explored indicates approximately the number of minutes the operator should devote to each region. Depending on the problem, there will be enough time to explore 4, 5, or 6 regions.

At the end of 15 minutes the computer will have stored:

1. The utility of the path comprised of the first way points and leg speeds.
2. The utility of each best-solution-to-date at the end of each minute.
3. The utility of the most recent solution at the end of each minute.

These are the data that are used in the analysis of operator generated data.

B. DYNAMIC PROGRAMMING OPTIMIZATION

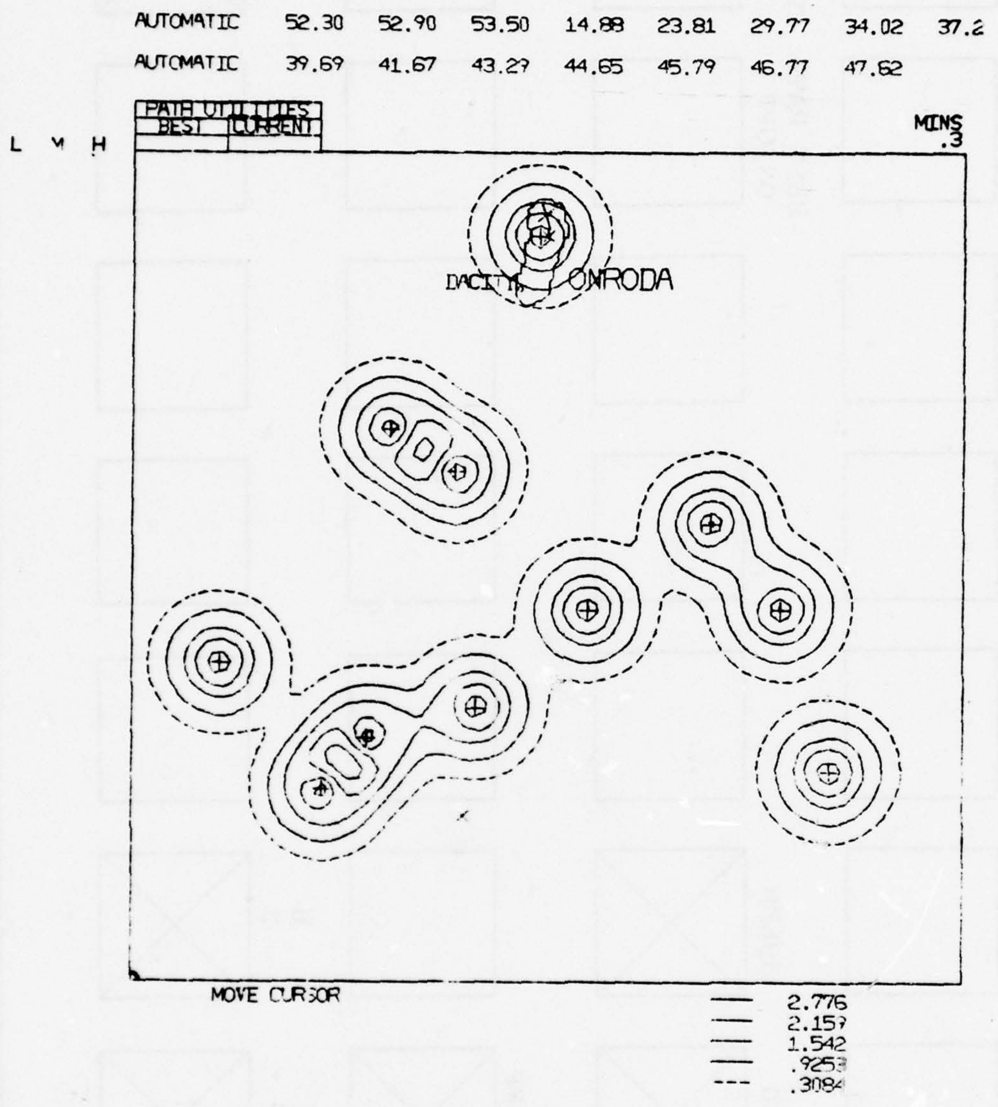
1. Characteristics of the Dynamic Programming Optimization Techniques Used in the Experiment

A dynamic programming (DP) optimization technique is used in the experiment. The "setup" for using the dynamic programming technique includes a grid network of evenly spaced points and a choice of three aircraft speed levels, namely, low, medium, or high. The DP technique specifies the best path by connecting points on the grid between air strike launch point and target and specifying one of the three speeds for each path leg between two connected points. The advantage of DP is that it does find the best path for the grid and speed levels it is using. A disadvantage of DP is that it takes a long time (even with the computer doing the number crunching) to reach a solution. For example, the time to reach a solution for a nine-by-nine grid with three speed levels is about four minutes; it is about eight minutes for a ten-by-ten grid. This occurs because DP investigates every allowable solution and then picks the best. If the grid size has finer resolution, e.g., 18 x 18 or 36 x 36, or the number of allowable speeds is larger than three, then solution time increases greatly. Another disadvantage is that the coarse grid points are often unpropitiously located with respect to the detection capability contours. The result is that the algorithm will avoid some valleys in the detection contours because there are no grid points in the valleys.

The DP algorithm uses the classical, grid-oriented, backward recursion approach. Minor adaptations have been made to allow the classical approach to accept the nonlinear utility criterion function. These are thoroughly discussed in Appendix C.

2. Operator Aided Optimization Using the Dynamic Programming Optimization Algorithm

The display appears as shown in Figure 10 at the beginning of a problem to be solved with the DP algorithm. The operator uses the function button box (see Figure 11) and the track ball to tell the program what to consider when it works on the problem. Operator inputs include:



```
AUTOMATIC   52.30   52.90   53.50   14.88   23.81   29.77   34.02   37.2
AUTOMATIC   39.69   41.67   43.27   44.65   45.79   46.77   47.62
```

Figure 10. Display Appearance at Beginning of Problem to be Solved Using DP Algorithm.

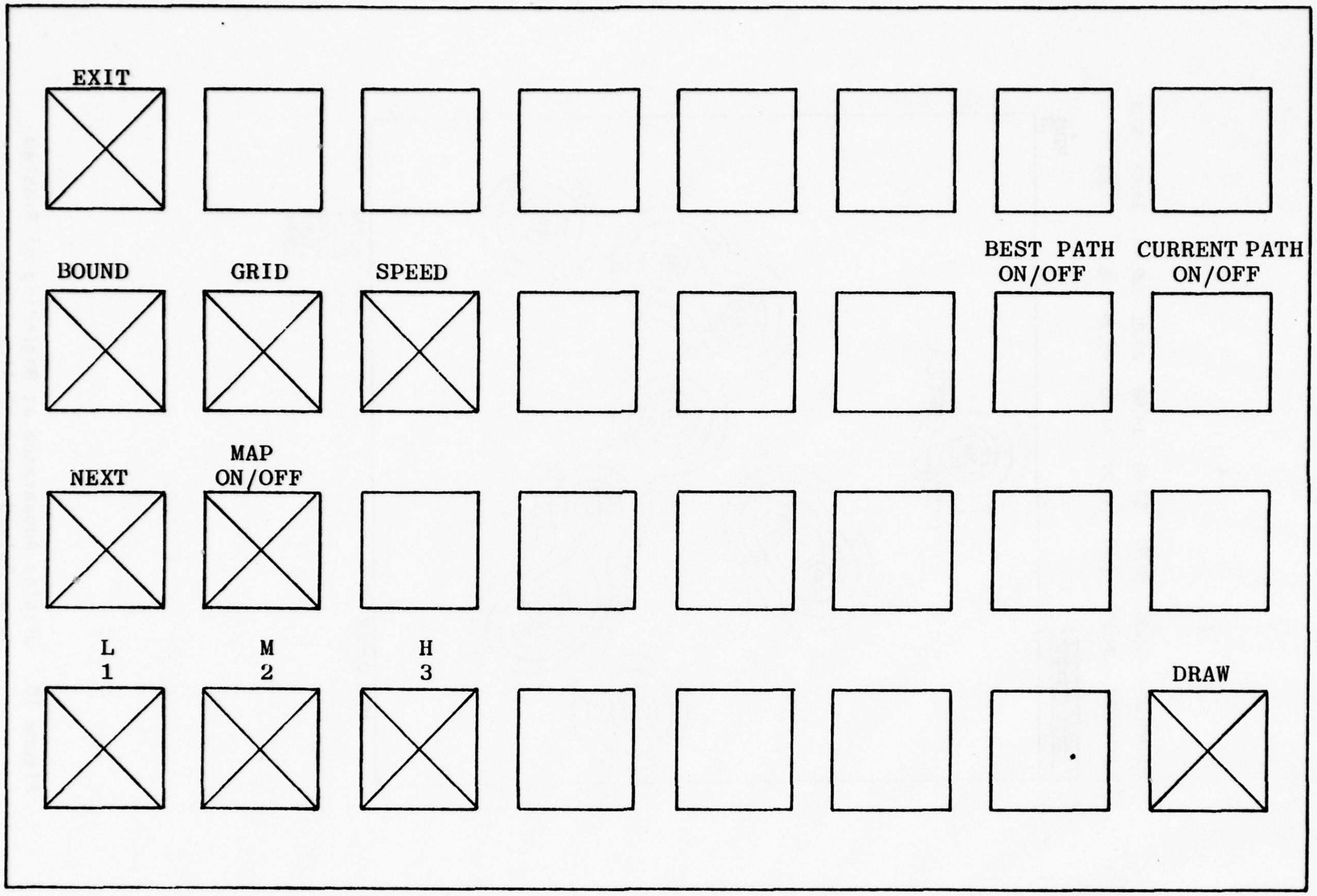


Figure 11. Function Button Configuration for the Dynamic Programming Optimization.

1. A boundary drawn around a region. This algorithm confines its search for a solution to this region.
2. The grid size to be used by the algorithm, that is, 9 x 9, 18 x 18, or 36 x 36.
3. Speeds that the algorithm is <u>not</u> to consider when searching for a solution.

In most cases, the algorithm finds a trial path considering the operator inputs, displays this path and its utility, and stops. (The exception is covered below.) It does not start again until the operator has completed a new set of inputs or tells the algorithm to do another iteration using the old inputs. New inputs may be 1, 2, and 3 above, or 2 and 3, or 3 only.

The operator's first task is to draw his estimate of the best path and write his estimate of the best leg speeds. He does this on a separately provided paper copy of the problem. This solution represents the Operator Unaided concept.

The operator's next task is to decide in his mind the rough outlines of the regions he will want the algorithm to explore. In general, he will pick two or three large regions and use the coarsest grid size, namely, 9 x 9, to explore these. In the beginning, a boundary should be drawn so that it encompasses more than one viable path. This way the solution provided by the DP algorithm reduces the ambiguity about where the better paths reside. When he finds the region that has the best path, he refines his solution by using a finer grid size (normally 18 x 18) and making the area within the boundary smaller.

When the operator has decided the first region he wants to explore, he responds to the flashing prompt MOVE CURSOR at the lower left of the display by moving the track ball to draw the boundary. This is indicated by the fact that the function button DRAW in Figure 11 is lighted. The launch point and the target must be contained within the closed boundary. If one of these points is not within the boundary, the computer will recognize this as an error, erase the boundaries drawn, and give the prompt MOVE CURSOR again. Thus, if the operator wants to redefine his boundary after some of it has been drawn, he can start over by closing the boundary without including the launch or target points.

When the boundary has been closed, the computer is ready to accept the specification of grid size as indicated by the flashing prompt SELECT GRID and the three lighted function buttons marked 1, 2, 3 at the lower left of the box. By pushing 1, the coarse grid is selected and displayed. Pushing 2 selects the medium grid of 18 x 18, and pushing 3 selects the fine grid of 36 x 36.

Now the prompt BOUND SPEED flashes at the lower left of the display. The arrow at the top left points to a space between two adjacent horizontal grid rows. By now pushing the buttons L, M, or H at the bottom left of the function box, the operator deletes from consideration by the algorithm low, medium or high speed for any path leg that crosses between the two rows and for any horizontal leg in the upper of the two rows. When the operator is finished specifying speeds to be deleted from consideration, he pushes the lighted button NEXT. Upon doing this, the arrow moves down to the next pair of rows and the operator repeats the procedure. For example:

1. If the operator has pushed L, M, and NEXT, the algorithm will not consider low and medium speeds for any leg crossing between the two rows on either side of the arrow and for any horizontal leg in the upper row.

2. If the operator has pushed only NEXT, then the algorithm will not delete any speeds and the arrow will move down to the next pair of rows.

3. If the operator pushes L, M, <u>and</u> H for any pair of grid rows above the target or below the launch point, the algorithm will not consider any leg that would be above the target or below the launch point. In this case, deleting L, M, and H is permissible and will save time.

4. If the operator pushes L, M, <u>and</u> H for any pair of grid rows between the target and launch points, the computer will be unable to find any solution. At the end of several minutes, the display picture will reappear and the blinking prompt NO SOLN YET will appear at the lower left of the display. In this case, deleting L, M, and H is not permissible and wastes time.

The operator continues to delete speeds as he desires until either (a) he has deleted speeds for the last pair of rows or (b) he decides that he does not want to delete any more speeds. In either case he then pushes the EXIT button. This completes the operator's input and the display screen goes blank while the computer is working on the trial solution. This may take two to five minutes depending on the size of the region within the boundary, the grid size, and the number of speeds deleted.

In most cases when the display reappears, the trial solution path is shown and the utility for that path and the function buttons BOUND, GRID, SPEED and NEXT are lighted. The operator does not know if the trial solution is the best possible solution for his inputs or not. He has two basic choices:

1. If there are two trial paths already displayed, he remembers the utility value for the most recent iteration displayed in the box at the top center of the display and then pushes NEXT. By pushing NEXT, the algorithm will perform another iteration and show the new and best-to-date trial solutions and their utilities. If the remembered utility of the previous trial and the utility of the current trial are the same, then the optimum solution for the inputs has been found. If there is little difference in the two most recent utilities, then the best tactic will usually be to start investigating another region or to refine the solution in the current region by using a finer grid.

2. By pushing BOUND, GRID, or SPEED he can redefine the inputs considered by the algorithm. If he pushes BOUND, then he must go through all three steps of drawing the boundary, selecting grid size, and deleting speeds. If he pushes GRID, then the algorithm will use the previously drawn boundary and the operator selects grid size and deletes speeds. If he pushes SPEED, the algorithm accepts the previously drawn boundary and grid and the operator only deletes speeds. If the area within the boundary was large, then the operator should redefine the boundary to include a much smaller number of points in the vicinity of the path selected by the previous iteration.

If the display reappears without a new trial solution (the exception previously noted), NO SOLN YET will flash at the lower left of the display. This means that the algorithm has not been able to find a complete trial solution on a single iteration. In this case the operator's suggested response is to push the NEXT button so that the algorithm will go to the next iteration to complete the trial solution.

IV. DESCRIPTION OF THE EXPERIMENTS

A. EXPERIMENTAL DESIGN

1. Hypotheses

Two types of experiments were conducted. In the Type I experiment, the performance of unaided operators was compared against the performance of the same operators using an optimization aid. In the Type II experiment the performance of operators using an optimization aid was compared against the performance of an automated optimization procedure as a function of time. These two types of experiments were conducted for each optimization algorithm, namely, the nonlinear programming (NP) algorithm and the dynamic programming (DP) algorithm.

The experimental null hypotheses tested for the NP algorithm were:

1. Path utilities generated by operators are not significantly different as a function of concept (unaided versus OAO), prior experience using the DP aid, operators, replications, or their interactions.

2. Path utilities generated by operators using OAO are not significantly different from path utilities generated by the automated NP algorithm using the random starting point (see Subsection III-A for the definition of random starting point).

3. Path utilities generated by operators using OAO are not significantly different from path utilities generated by the automated NP algorithm using the parabolic starting point (see Subsection III-A).

The experimental hypotheses tested for the DP algorithm were:

4. Path utilities generated by operators are not significantly different as a function of concept (unaided versus OAO), prior experience using the NP aid, operators, replications, or their interactions.

5. Path utilities generated by operators using OAO are not significantly different from path utilities generated by the automated DP algorithm.

2. Independent Variables

The independent variables for the Type I experiments were:

1. System concepts
2. Prior experience using OAO
3. Operators
4. Replications

The independent variable for the Type II experiments was system concept, namely operator aided optimization versus automated optimization.

a. Prior OAO Experience. Eight of the 16 operators worked the DP problems first and then the NP problems. The other eight operators did the NP problems first. Thus for one set of data, for example the NP data, half the data was generated by operators with no prior experience using either the NP or DP aid and half was generated by operators with prior experience using the DP aid.

b. Operators. There were 16 operators. Descriptive information about the operators and their training is given in Subsection IV-B.

c. Replications. Each operator was given a set of 12 problems to be solved using the appropriate OAO procedure (one set of 12 for the NP aid, another set of 12 for the DP aid). At the beginning of each problem the operator recorded his estimate of the best solution. Then he proceeded to use the OAO procedure. All subjects worked the same set of twelve problems. The learning effect was tested by comparing performance on the first six problems against performance on the last six problems. Thus, one replication for operator generated data consisted of six problems.

The automated NP concept does use random numbers to generate starting points. For this reason each of the 12 NP problems was run five times using a different random number seed for each trial.

d. <u>System Concepts for Type II Experiments</u>. The system concepts for the NP algorithm were operator aided optimization, automated optimization using random starting points, and automated optimization using parabolic starting points. The system concepts for the DP algorithm were operator aided optimization and automated optimization.

3. Dependent Variables

a. <u>Type I Experiments</u>. The dependent variable for the Type I experiments comparing unaided optimization with OAO was normalized utility. The raw data for unaided optimization was the utility of the path selected by the operator. The raw data for OAO was the utility of the best path found by the operator using OAO during the fifteen-minute trial. For each problem these data points were normalized by dividing each value by the highest utility achieved by any operator or the automated algorithm on that problem. Thus hypotheses 1 and 4 were tested by comparing normalized utility of the unaided solution against the normalized utility of the best OAO solution.

b. <u>Type II Experiments</u>. The raw data for OAO and automated optimization at time "t" was the best utility found by the operator or automated algorithm from time zero to the end of time "t". Table 3 illustrates the meaning. The raw data across operators and the automated algorithm was normalized for each minute of each problem by the best utility achieved by any operator or any run of the automated algorithm.

Table 3. Example of Data Generated During an OAO or Automated Optimization Trial.

Minute	Best Utility Generated During the xth Minute	Raw Data, i.e., Best Utility from Time Zero to End of xth Minute
1	20	20
2	60	60
3	62	62
4	45	62
5	54	62
6	55	62
7	49	62
8	65	65
9	67	67
10	68	68
11	37	68
12	52	68
13	56	68
14	57	68
15	30	68

If raw utility data achieved by two competing methods A and B are as shown in Figure 12 (a), then it is clear that method A is superior. However, if utilities achieved are as shown in Figure 12 (b), then the time preference for utility must be stated so that the optimizers (operator or automated algorithm) may use search strategies best suited for the stated preference.

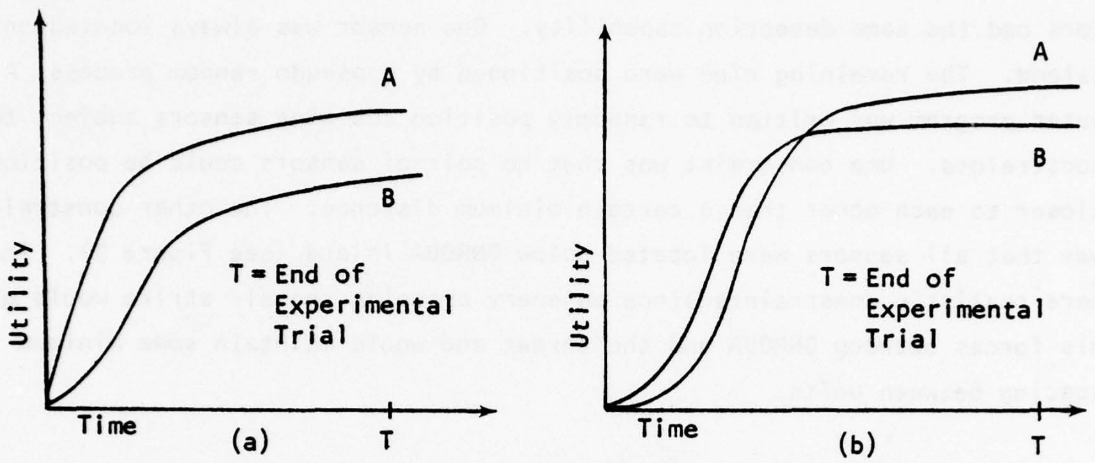


Figure 12. Hypothetical Utilities Achieved by Competing Methods.

Two dependent variables were used to test hypotheses 2, 3, and 5. One was the best utility to date (since time zero). The other was the time average of best utility to date according to the formula:

$$\overline{U}(t) = \frac{1}{T} \sum_{t=1,2,\ldots}^{t} U(t) \qquad (9)$$

where U(t) is the normalized utility at time "t" of the best utility to date. With these dependent variables it was possible to test hypotheses 2, 3, and 5 for:

1. Any specific time
2. Any specified time interval by comparing data curves.

4. Problem Variables

The elements that defined a given problem were the adaptation of the ONRODA airstrike scenario map (Section II-A), sensor locations, and strike path start point. The steps described below were taken to make problems nearly equally difficult for the operators.

Each problem used the same number of sensors, namely, ten, and all sensors had the same detection capability. One sensor was always located on ONRODA Island. The remaining nine were positioned by a pseudo random process. A computer program was written to randomly position the nine sensors subject to two constraints. One constraint was that no pair of sensors could be positioned closer to each other than a certain minimum distance. The other constraint was that all sensors were located below ONRODA Island (see Figure 7). These were realistic constraints since an enemy opposing the air strike would group his forces between ONRODA and the threat and would maintain some minimum spacing between units.

About 50 configurations of sensors generated by the program were examined by the experiment designers. Starting points for the air strike were manually selected so that the largest number of paths having nearly equal utility would result for each of the 50 problems. Then the 24 "best" problems were selected as experimental problems. The basis for selecting the experimental problems was (a) at least three paths having nearly equal utility and (b) no path selection strategy was best for a large majority of the problems. (In the previous experiment reported in Reference 1, the strategy of selecting paths along the right or left edge of the geographical area displayed to the operator was best for nearly all problems.) Thus, problem difficulty was not treated as an independent variable in the experiment because:

1. Problems were constructed to be nearly equally difficult.
2. Normalization of raw data tends to eliminate whatever differences in problem difficulty remained after the problems were selected.

5. ANOVA Design for Comparing Performance of Unaided Operators with OAO

The purpose of the Type I experiment was to determine if the operators would achieve better performance using the NP and DP aids for 15 minutes than they would achieve without the aids. A nested factorial, randomized block experiment was conducted. The factors were:

- Concepts (C_i) - 2 levels (Unaided operator and operator aided optimization)
- Prior Experience (P_j) - 2 levels (Half the operators did DP problems first and NP next; the other half did NP, then DP)
- Operators ($O_{k(j)}$) - 8 levels nested within training; therefore 16 operators total
- Replications (R_l) - 2 levels (First half of trials and second half).

There were no designed differences in problem difficulties. Thus, differences in problem difficulties were not treated as a factor. Any spurious differences were mitigated by (a) using normalized data in the analysis and (b) balancing problems within each group of 8 subjects so that 4 subjects got one set of 6 problems in the first replication and the other 4 subjects got the remaining 6 problems in the first replication. The model for the normalized dependent variable is:

$$Y_{ijklm} = \mu + C_i + P_j + CP_{ij} + O_{k(j)} + CO_{ik(j)} + R_l + CR_{il} + PR_{jl} + CPR_{ijl} + OR_{k(j)l} + COR_{ik(j)l} + \varepsilon_{m(ijkl)} \quad (10)$$

B. OPERATORS AND TRAINING OF OPERATORS

There were three subgroups within the 16 operators. Ten operators were juniors or seniors from UCLA majoring in engineering or computer science. Two operators were the staff members at Integrated Sciences who designed the aids and the experiment. The remaining four operators were alike in that none of them had a technical background. Their educational backgrounds were:

- Master of Arts (Education)
- Master of Arts (English)
- Master of Arts (Urban Affairs)
- Associate of Arts (Art)

Operator training for use of each aid, i.e., the NP aid and the DP aid, was conducted in three phases: orientation, exercise using training problems, and a pre-experiment briefing. Orientation for each aid began with reading the training materials developed for that aid. The training materials treated the following topics:

- Purpose of the experiment
- Representation of sensor detection capability on the display
- The utility function
- Characteristics of the optimization technique used (NP or DP)
- Operation of the aid
- Example of a problem worked out (five figures for the DP aid, nineteen figures for the NP aid, with text comments and guidelines accompanying each figure).

The training materials are in Appendices D and E.

After each operator read the training materials, he conferred with one of the ISC staff members who designed the experiment. Operator questions were answered during this conference and the ISC staffer verbally tested the operator's understanding of the problem situation and use of the aid. Then the operator worked eight problems at the display. Questions that arose during these problems were answered by an ISC staff member with experience using the aid. Another conference between the operator and one of the experiment designers was held after the training problems were completed. This conference focused on the search strategies that the operator had learned and planned to use. The experiment designer advised the operator of potential pitfalls and useful modifications to the planned strategies. The operator began his experimental trials after this conference. No further training was given during the trials. The average training time across operators was about four hours for each aid.

C. EXPERIMENTAL PROCEDURE

The experimental team consisted of the test director and an operator. Each operator was assigned a unique identification code and the sequence of the twelve problems corresponding to that code was stored in the computer. The test director scheduled the software and entered the operator's code. That procedure "brought up" the operator's next (uncompleted) problem on the display. At the beginning of the problem the operator entered his best estimate of the solution as described in Sections III-A-3 for the NP aid and III-B-2 for the DP aid. Entering this best estimate took between 0.8 and 1.2 minutes for the NP aid and 0.5 and 0.9 minutes for the DP aid. (These estimates are based on the personal experience of the experiment designers and their observations and conferences with operators.)

Feedback to the operator on his performance was provided throughout each trial. This was done in the following way: the time averaged performance of the automated algorithm was displayed at the beginning of the trial for all 15 trial minutes. This display was located just above the picture of the scenario. At the end of each minute the computer calculated and displayed the operator's time averaged performance under the corresponding value for the automated algorithm. Thus the operator obtained feedback on how he was doing in an absolute sense and in a relative sense compared to his "competition," the automated algorithm.

The test director remained on call during each trial to monitor the trials, troubleshoot any equipment malfunctions or operator-induced problems in entering path data, and to bring up the next trial once the previous trial was completed. The test director spent part of the time in the computer and display facility where the operator worked the problem and the remaining time in an adjacent room. Operators normally did two, three, or four trials in a row before taking a break. Multiple trials were permitted because operators did not experience fatigue after as many as three sequential trials.

V. RESULTS

A. ANALYSIS OF UNAIDED OPERATOR AND OAO DATA FOR NP AID

A four-way analysis of variance was performed on the normalized path utility data generated by operators with and without the NP aid. After pooling procedures (Reference 5) were applied, the ANOVA results were as shown in Table 4.

The operators using the aid did significantly better than without the aid. This was expected since they had 15 minutes to improve the solution by using the aid. Of greater interest is the degree of improvement made possible with this aid. The average improvement across all subjects and trials was 29% with a range of 9% to 123%.

Performance was significantly different across operators but this was solely due to performance variability in unaided operation. This can best be understood by considering the following: average normalized utility without the aid was 77.24 utility points; the standard deviation was 13.54. Average normalized utility with the aid was 99.5; the standard deviation was only 0.38. Thus the aid served as an "equalizer." It enabled operators having relatively low utility scores without the aid to do as well when using the aid as those who had relatively high scores without the aid. The lack of a technical education was apparently not an impediment to good performance with or without the aid. Figure 13 shows the average utility with and without the aid for the 12 operators who have a technical education and the four operators who don't.

The significant interaction between prior experience and replications occurs solely on unaided operator data. The interaction is plotted in Figure 14. The plot indicates that the prior experience using the DP aid was a handicap rather than a help. However, the performance gap was smaller in the second replication than it was in the first.

Table 4. Analysis of Variance for Comparing Unaided Operator and OAO with the NP Aid.

Source of Variation		Degrees of Freedom	Sum of Squares	Mean Square	F_{obs}
Concepts	(C)	1	47,597	47,597	327.3*
Prior Experience	(P)	1	2,745	2,745	4.6
C x P		1	2,580	2,580	5.0
Operators	(O)	14	14,248	1,018	4.1*
C x O		14	13,451	961	4.2*
P x Replications		1	598	598	4.1*
C x P x Replications		1	517	517	3.6*
O x Replications		14	3,514	251	1.7*
C x O x Replications		14	3,206	229	1.6*
Pooled Error		322	46,829	145	-
Totals		383	135,285		

*α = 0.10

Table 5. Analysis of Variance for Comparing Unaided Operator and OAO with the DP Aid.

Source of Variation		Degrees of Freedom	Sum of Squares	Mean Square	F_{obs}
Concepts	(C)	1	9,514	9,514	48.2*
Prior Experience	(P)	1	3,752	3,752	19.0*
C x P		1	496	496	1.2
Operators	(O)	14	20,691	1,478	7.5*
C x O		14	3,038	217	1.1
C x P x Replications		1	424	424	2.2
Pooled Error		351	69,246	197	-
Totals		383	107,161		

*α = 0.10

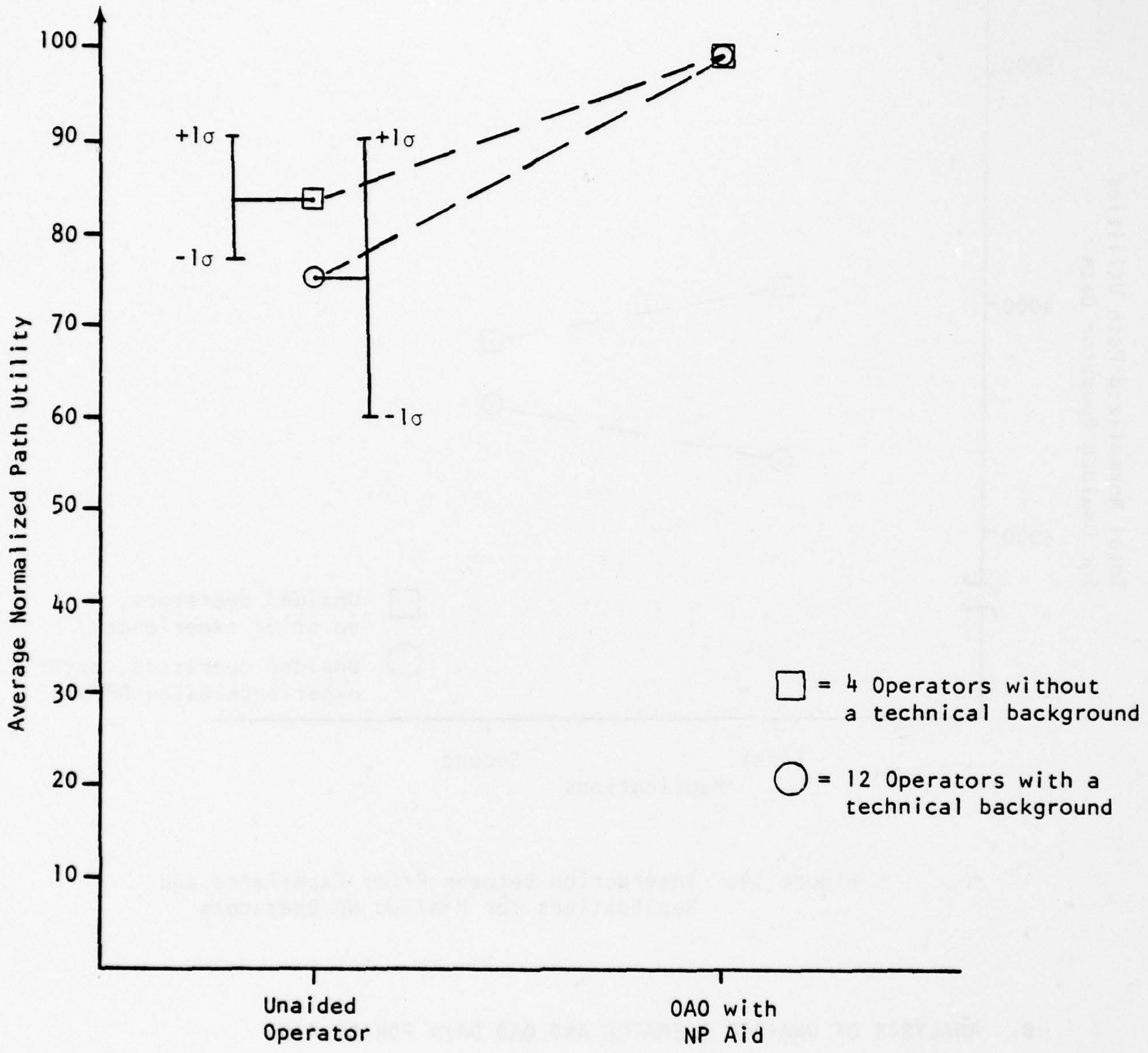


Figure 13. Average Performance of NP Operators With and Without a Technical Background.

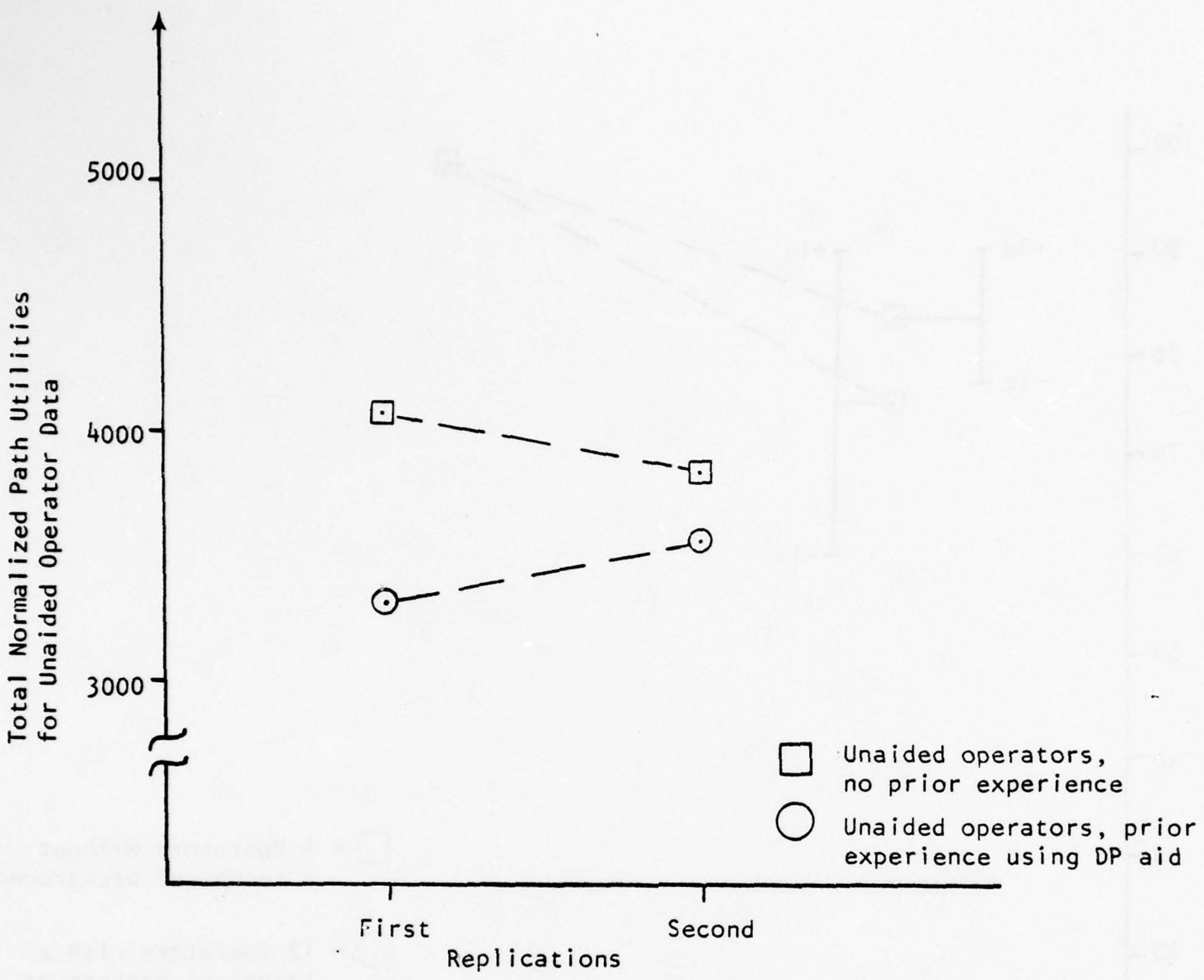


Figure 14. Interaction Between Prior Experience and Replications for Unaided NP Operators

B. ANALYSIS OF UNAIDED OPERATOR AND OAO DATA FOR DP AID

A four-way analysis of variance was performed on the normalized path utility data generated by operators with and without the DP aid. After pooling procedures were applied, the ANOVA results were as shown in Table 5.

Operators using the aid did significantly better than without the aid. Again, this was expected since they had 15 minutes to improve the solution using the aid. The average improvement across all subjects and trials was 12% with a range of -9.1% to 27%. The lower number in the range was due to two trials by one subject who did not sufficiently constrain the problem. Consequently, on one trial the algorithm never reached a solution within 15 minutes, and on the other trial it yielded a normalized utility of only 20.4 near the end of the 15-minute period. These trials did not represent the operator's steady state performance and thus they were eliminated from the following analysis. With the two unrepresentative trials eliminated, the lower end of the range was 3.5% improvement.

Performance was significantly different across operators but there was no interaction with concepts (unaided versus OAO) as was the case with the NP data. Average normalized utility without the aid was 80.3 utility points; the standard deviation was 9.8. Average normalized utility with the aid was 91.2; the standard deviation was 7.0. Again, the lack of a technical education was apparently not an impediment to good performance with or without the aid. Figure 15 shows the average utility for operators with and without a technical education.

Operators who had previously used the NP aid did significantly better than those without that experience. The difference on a percentage basis was 7.6%.

C. ANALYSIS OF OAO AND AUTOMATED RESULTS WITH THE NP ALGORITHM

The previous subsections treated data generated by operators. Comparisons were made between:

1. Utility generated at the beginning of a problem when the operator estimated the best path without any aid, and
2. The best utility the operator was able to generate using the aid over a period of 15 minutes.

This and the next subsection compare as a function of time the performance of operators using an aid with automated performance of the same algorithm which is controlled by the operator during OAO.

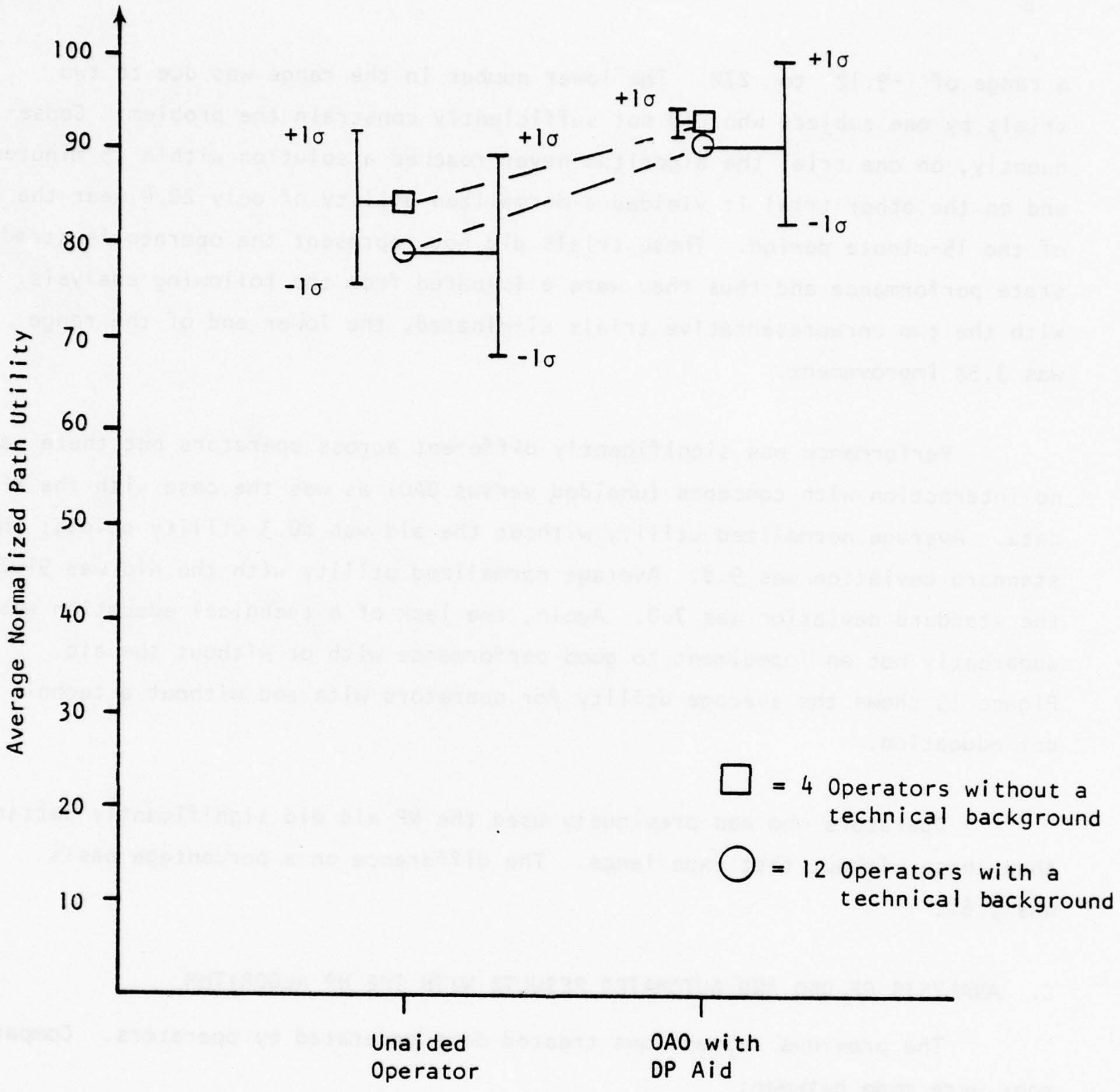


Figure 15. Average Performance of DP Operators With and Without a Technical Background.

Figure 16 shows a plot of best-to-date normalized utility versus time for OAO and automated use of the NP algorithm. All three curves are significantly different from each other at the 10% level by the Wilcoxon matched-pairs, signed-ranks test. The plot verifies that the parabolic starting point procedure is clearly superior to the random starting point procedure at all times. Superiority ranges from 215% at minute 1 to 12% at minute 15. OAO using the NP algorithm is superior to automated use of the algorithm with parabolic starting point; superiority is 27% at minute 1, 53% at minute 2, and then gradually decreases to 27% at minute 15. The average utility accomplished by unaided operators is shown as a small square plotted at 1.0 minute, the subjectively determined average time when the operator's unaided estimate was input. The reader can compare the gradual improvement in the average OAO solution with the average unaided solution.

Figure 17 shows a plot of time averaged normalized utility calculated according to the scoring rule (Equation 9) versus time for OAO and automated use of the NP algorithm. Again, all three curves are significantly different from each other at the 10% level by the Wilcoxon matched-pairs, signed-ranks test. The plot shows the same pattern as observed in Figure 16. The main difference is a greater margin of superiority for the parabolic starting point over the random starting point.

Figure 18 shows "instantaneous" normalized utility for the OAO and automated NP algorithm data. Instantaneous utility is defined to be utility of the most recent solution at the end of the n^{th} minute. The OAO data quickly reach and maintain high values because the operators are able to rapidly input good starting points. The automated starting points are relatively poor compared to the operator selected starting points and therefore the average automated solution at any point in time is relatively poor. Parabolic starting points are usually superior to random starting points. Thus, instantaneous utility generated from parabolic starting points is superior to utility generated from random starting points as shown in the figure.

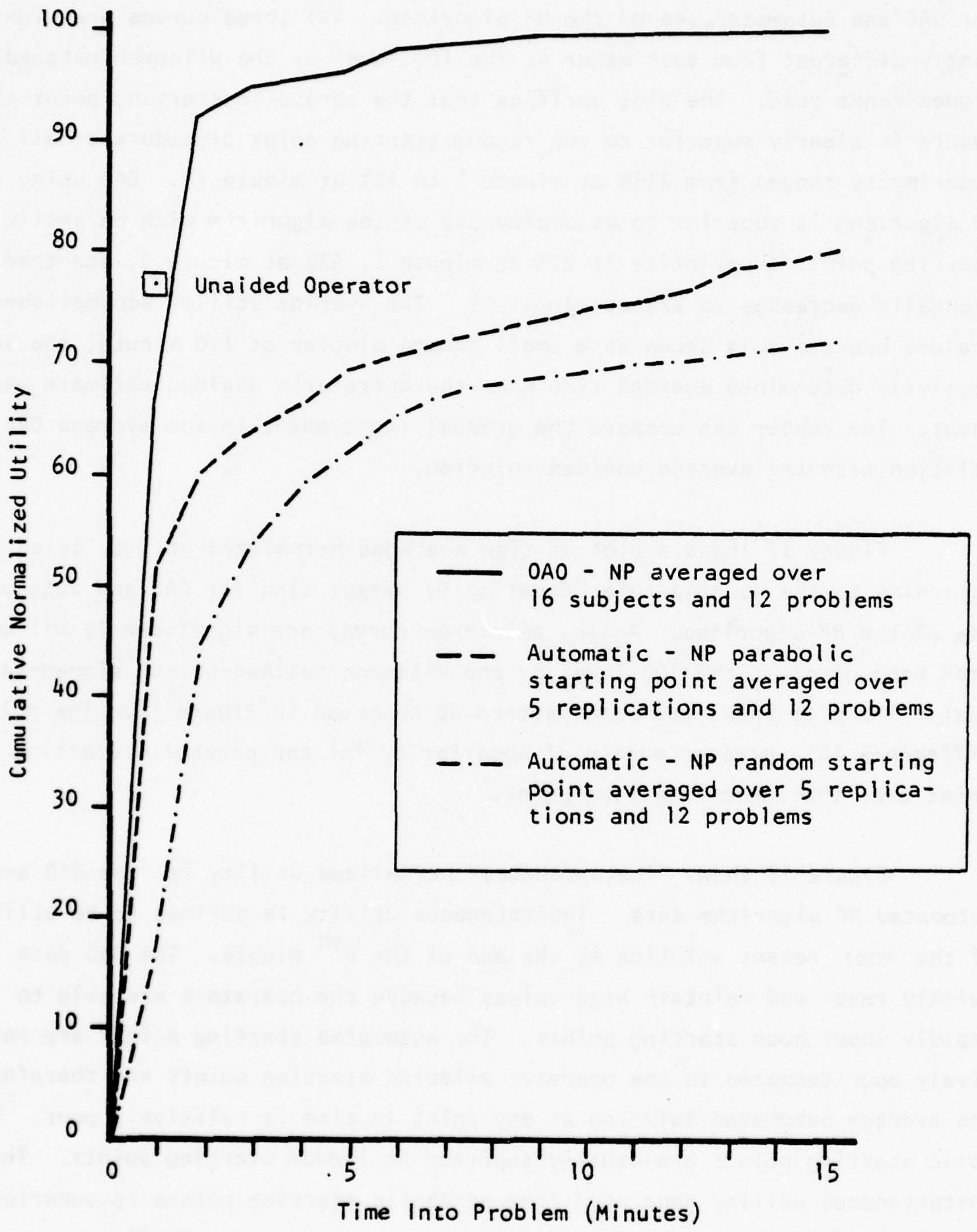


Figure 16. Best-to-Date Normalized Utility Versus Time for OAO and Automated NP Algorithm.

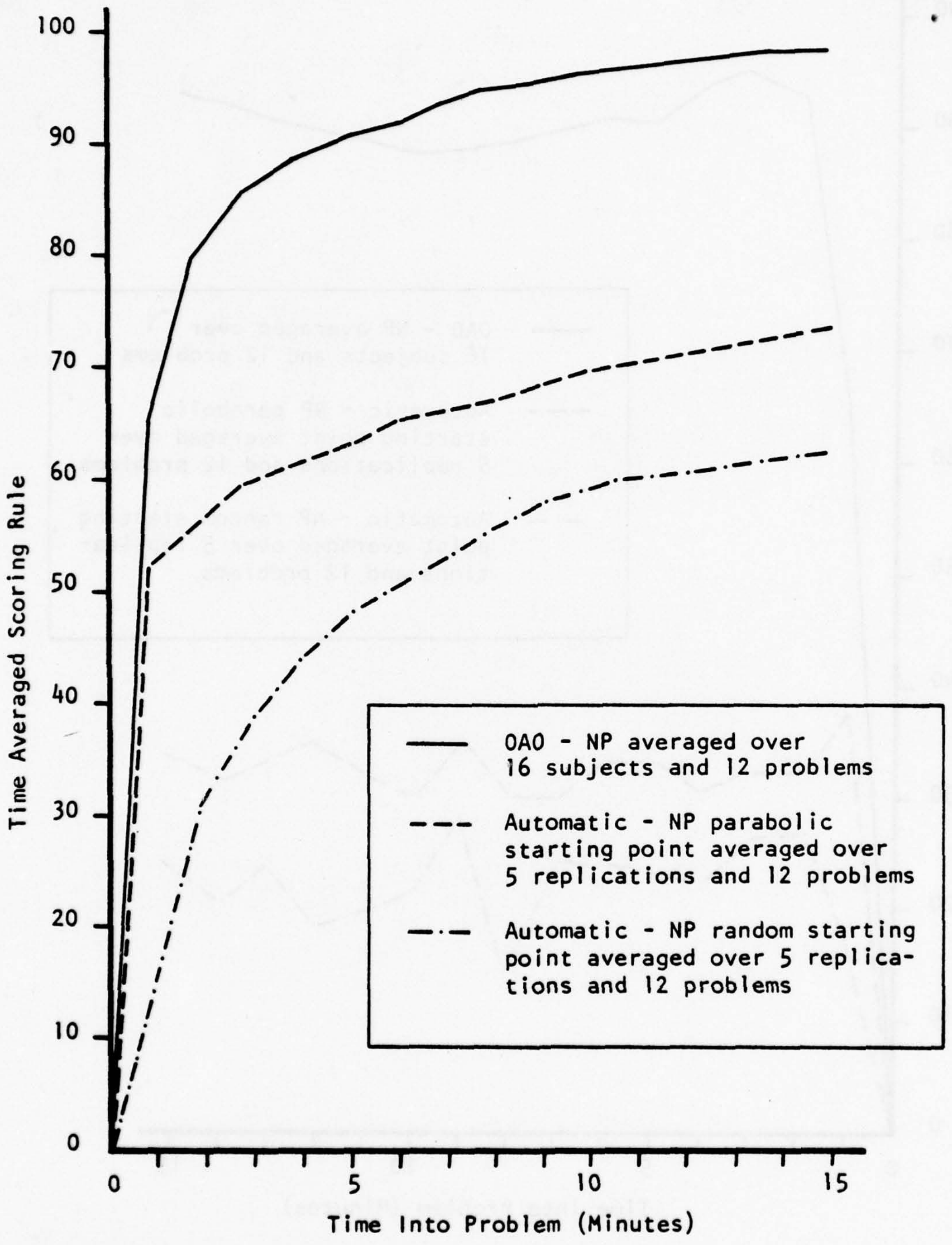


Figure 17. Time Averaged Scoring Rule Utility Versus Time for OAO and Automated NP Algorithm.

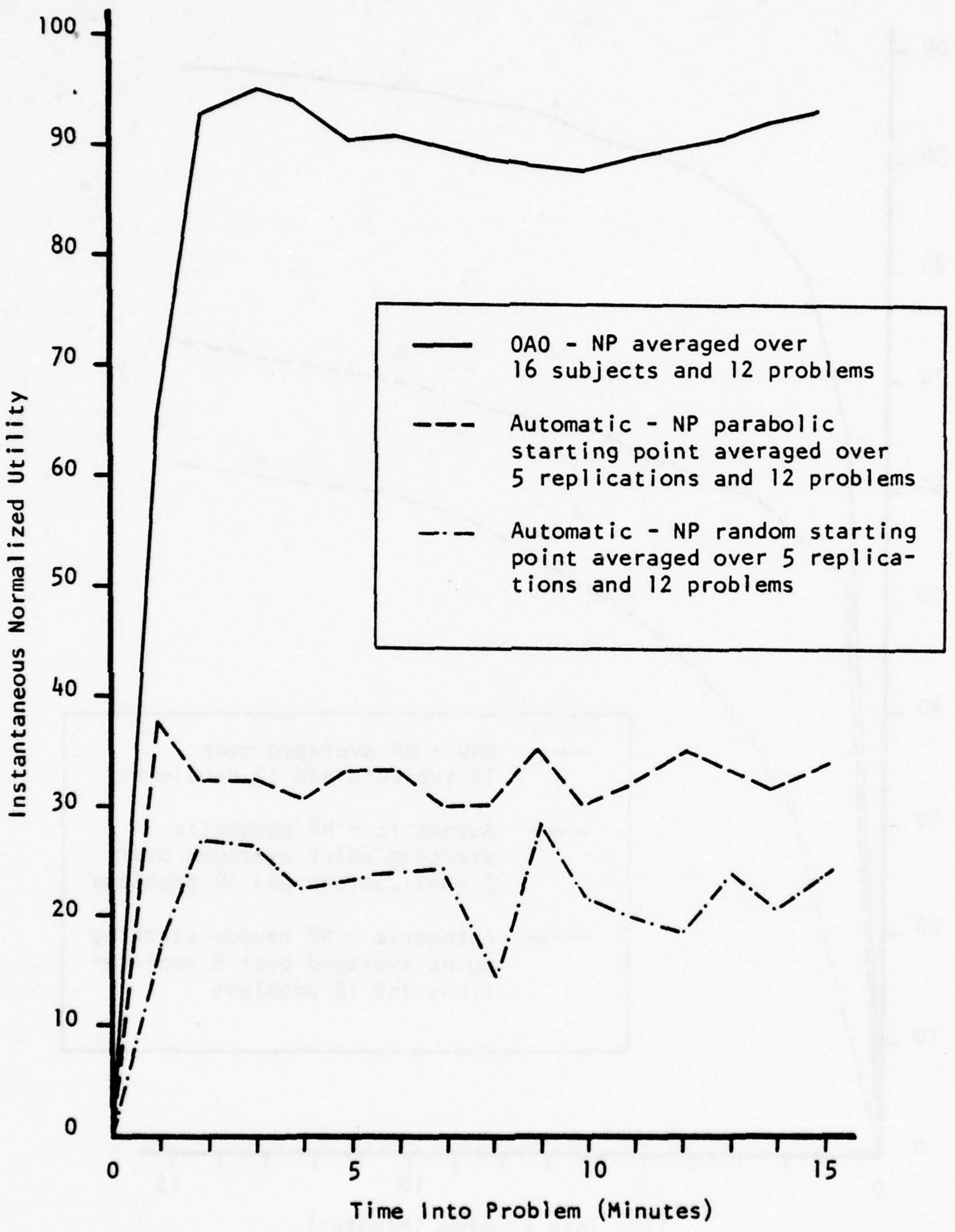


Figure 18. Instantaneous Utility Versus Time for OAO and Automated NP Algorithm.

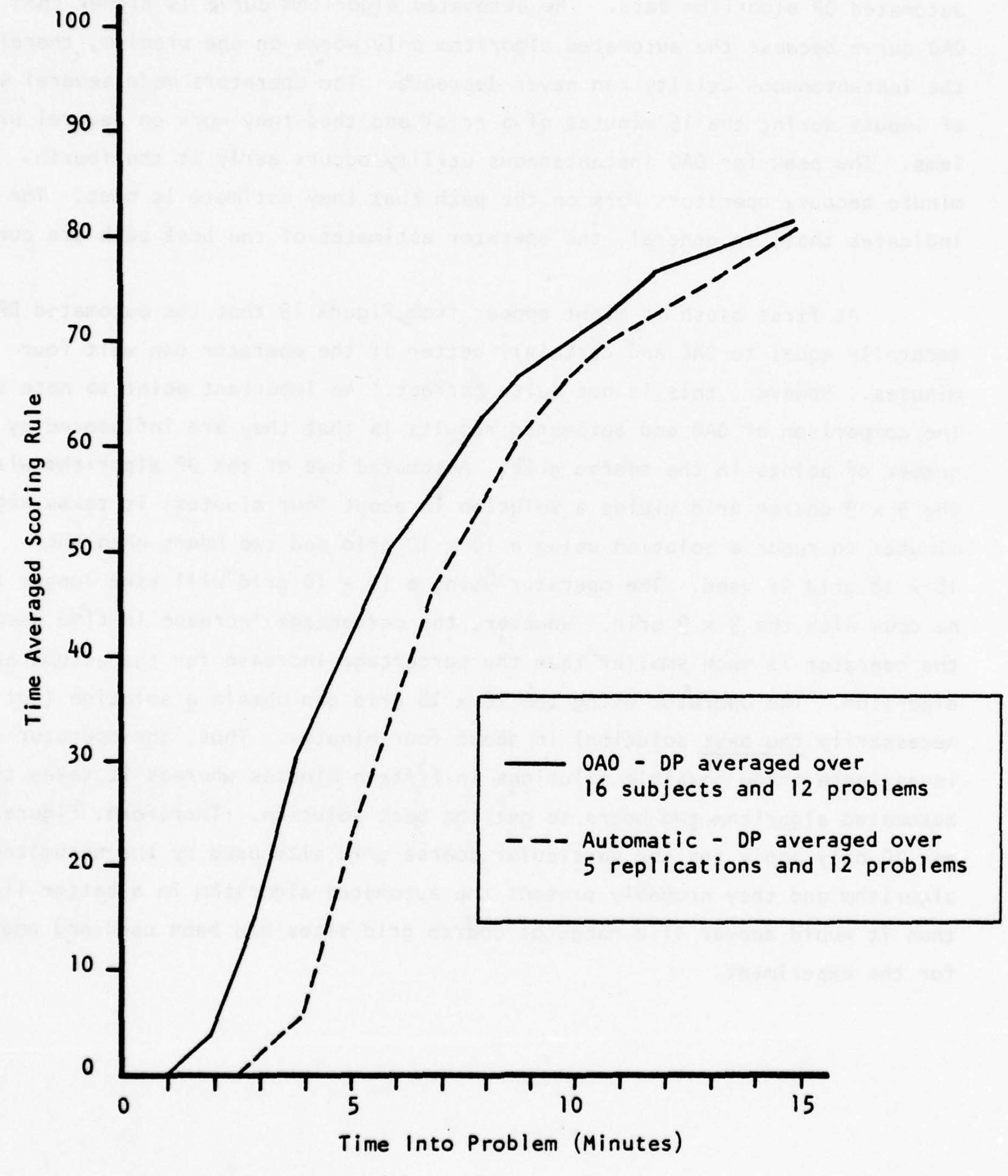


Figure 20. Time Averaged Scoring Rule Utility Versus Time for OAO and Automated DP Algorithm.

Instantaneous normalized utility is shown in Figure 21 for OAO and automated DP algorithm data. The automated algorithm curve is higher than the OAO curve because the automated algorithm only works on one problem, therefore the instantaneous utility can never decrease. The operators made several sets of inputs during the 15 minutes of a trial and thus they work on several problems. The peak for OAO instantaneous utility occurs early at the fourth minute because operators work on the path that they estimate is best. The peak indicates that, in general, the operator estimates of the best path are correct.

At first blush it might appear from Figure 19 that the automated DP is generally equal to OAO and certainly better if the operator can wait four minutes. However, this is not quite correct. An important point to note about the comparison of OAO and automated results is that they are influenced by the number of points in the coarse grid. Automated use of the DP algorithm with the 9 x 9 coarse grid yields a solution in about four minutes; it takes eight minutes to reach a solution using a 10 x 10 grid and two hours when the 18 x 18 grid is used. The operator using a 10 x 10 grid will take longer than he does with the 9 x 9 grid. However, the percentage increase in time used by the operator is much smaller than the percentage increase for the automated algorithm. The operator using the 18 x 18 grid can obtain _a_ solution (not necessarily _the best_ solution) in about four minutes. Thus, the operator can investigate three possible solutions in fifteen minutes whereas it takes the automated algorithm two hours to get the best solution. Therefore, Figures 19 and 20 only apply for the particular coarse grid size used by the automated algorithm and they probably present the automated algorithm in a better light than it would appear if a range of coarse grid sizes had been used and analyzed for the experiment.

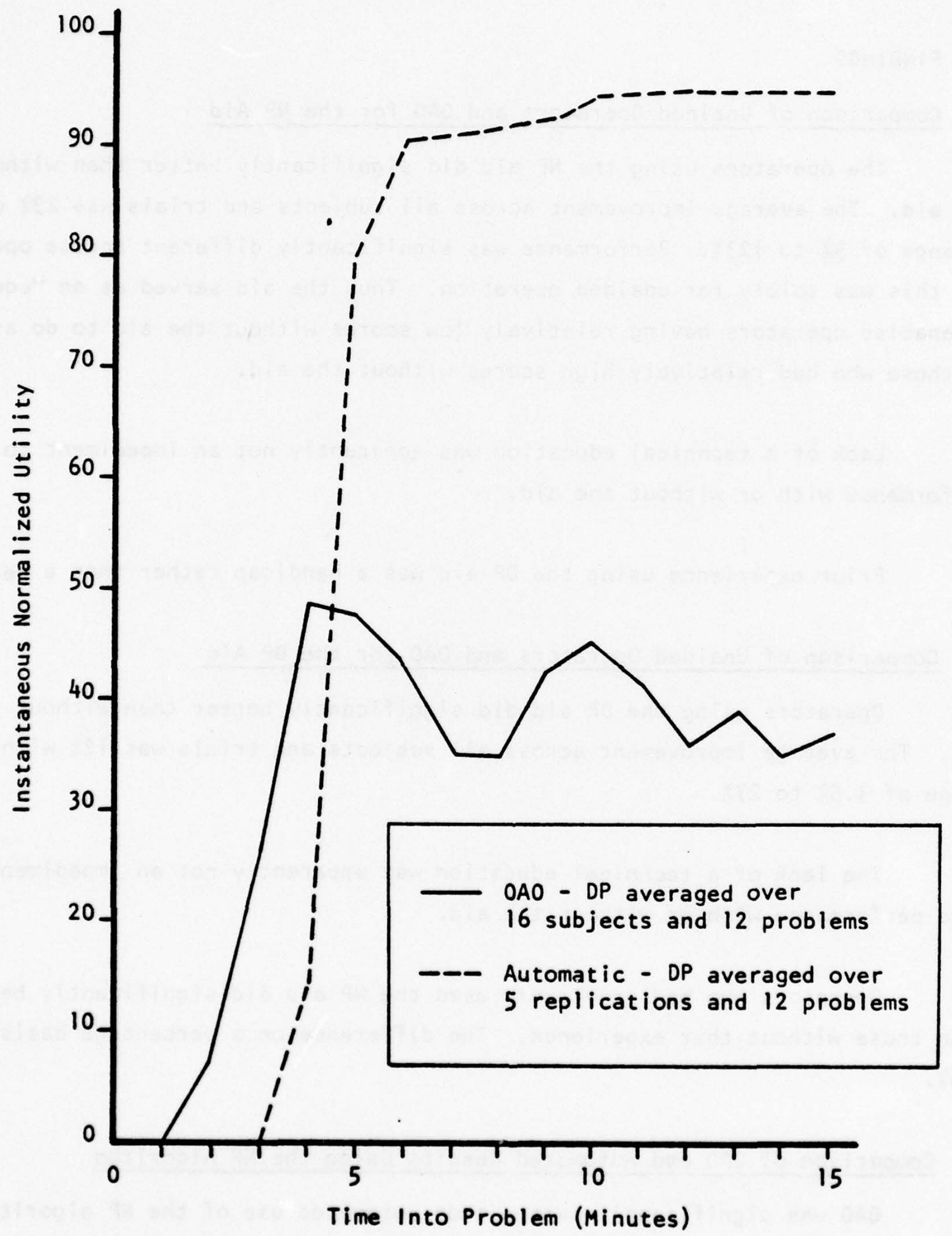


Figure 21. Instantaneous Utility Versus Time for OAO and Automated DP Algorithm.

VI. FINDINGS AND RECOMMENDATIONS

A. FINDINGS

1. Comparison of Unaided Operators and OAO for the NP Aid

The operators using the NP aid did significantly better than without the aid. The average improvement across all subjects and trials was 29% with a range of 9% to 123%. Performance was significantly different across operators but this was solely for unaided operation. Thus the aid served as an "equalizer." It enabled operators having relatively low scores without the aid to do as well as those who had relatively high scores without the aid.

Lack of a technical education was apparently not an impediment to good performance with or without the aid.

Prior experience using the DP aid was a handicap rather than a help.

2. Comparison of Unaided Operators and OAO for the DP Aid

Operators using the DP aid did significantly better than without the aid. The average improvement across all subjects and trials was 12% with a range of 3.5% to 27%.

The lack of a technical education was apparently not an impediment to good performance with or without the aid.

Operators who had previously used the NP aid did significantly better than those without that experience. The difference on a percentage basis was 7.6%.

3. Comparison of OAO and Automated Results Using the NP Algorithm

OAO was significantly better than automated use of the NP algorithm for both the Parabolic Starting Point and Random Starting Point rules. The Parabolic Starting Point rule produced significantly better performance than the Random Starting Point rule. Superiority of OAO over automated use of the

NP algorithm with the Parabolic Starting Point rule was 27% for best-utility-to-date data at minute 1, 53% at minute 2, and then gradually decreased to 27% at minute 15.

4. Comparison of OAO and Automated Results Using the DP Algorithm

Neither OAO nor automated use of the DP algorithm was consistently superior throughout the trial period for best-to-date utility data. OAO was superior to the end of the third minute. The automated algorithm was superior from minute 4 to minute 15 by margins ranging from 7% at the fourth minute, to 12% at the fifth minute, and then slowly decreasing to 5% at the fifteenth minute.

OAO was significantly better than the automated algorithm for time averaged data. However, the margin of superiority decreased from 92% at minute 4 and 41% at minute 5 to only 1% at minute 15.

These results apply for the 9 x 9 coarse grid size used by the automated algorithm. The results would be different if the number of points in the coarse grid were larger; it is highly likely that increasing the number of points in the coarse grid would favor OAO.

5. Miscellaneous Findings

The NP aid was less complex to use than the DP aid and operators generally preferred working with the NP aid to working with the DP aid. Operators using OAO with the NP aid found the global optimum on a higher percentage of trials than operators using OAO with the DP aid. The NP aid more naturally fits the air strike problem than the DP aid. The reason is that the NP aid operates on space and speed continuums whereas the DP aid must operate on discrete spatial grid points and speeds. Thus, using the DP aid for the air strike problem amounts to making a forced fit of a method that deals with discrete points to a problem described by continuous variables.

The average time required to adequately train an operator to use either aid was about four hours.

6. Potential Implication of the Findings

OAO is potentially attractive as a decision aid for a class of problems when all of the following are true:

1. Solution space has high dimensionality (e.g., ≥ 5).
2. Criterion function is nonlinear and multi-modal.
3. Pertinent problem definition information is not available with enough advanced warning to incorporate into the design of an operating optimization software package that would adequately handle all or most problems in the class.
4. Pertinent information concerning available decision options is also not available in time to impact software development
5. The problem can be represented in geometric/graphical format.

OAO is attractive when the above conditions hold because:

1. The operator can see what is happening during the optimization. With pictorial problem representation, he can make adjustments to the optimization procedure or results to compensate for limitations in problem representation more easily than he can when there is no pictorial representation.
2. The time required to train operators to use OAO with pictorial problem representation is apparently relatively short and does not require technical knowledge of the optimization algorithms.

B. RECOMMENDATIONS

1. OAO NP Versus OAO DP

If only one aid is to be implemented on the Operational Decision Aids facility at the University of Pennsylvania, then the NP aid should be chosen. If funds are available it would be worthwhile to implement the DP aid also so that Navy officers and R & D managers could get a feel for the way such an aid could be used. TFC decision problems that involve discrete variables should be examined for their applicability to an OAO DP type aid.

2. Taxonomy of Navy Command and Control Decisions

ISC found that the three-type characterization of machine participation in decision making used in this report was useful. Based on this usefulness, ISC suggests that the third type, man helping machine to help man, be used in future command and control studies. ISC will assume that command and control decisions can be categorized as ISC suggests. If this assumption is valid, then ISC recommends that decisions for which the third type is appropriate be further evaluated to determine if geometric/graphic representation is the most efficient way to represent the problem in each case.

3. Simulation of Real-Time Dynamics

The ocean-borne enemy sensors facing a real-world air strike planner are in motion during the planning and execution of the air strike. Consequently, the detection field representing the joint detection capability of enemy sensors is dynamic and not static. The problems used in the recently completed experiment show static sensors to the operators.

Representing the more realistic dynamic situation involves dynamically updating the detection contours to account for the changing positions of enemy sensors. The contour drawing algorithm for a machine stored analytic function representing joint sensor detection capability is relatively complex. In order to get the best "fit," i.e., best representation of the contours, it is currently necessary to vary a weighting factor, visually observe the contours drawn for each value of the weighting factor, and select the value that gives the best fit.

Recommended steps to be taken in implementing real time dynamics are:

1. Devise a method for automatically obtaining an acceptable fit for the contours.
2. Provide the operator with controls that will enable him to consider sensor movement while planning the air strike.

3. Provide the operator with controls that will enable him to consider sensor movement <u>during</u> the air strike and issue path change directions.

4. Design an experiment that would compare operator aided optimization performance with performance of an automated algorithm.

4. Implementation and Testing of a Gradient Search Algorithm for NP

A gradient-free algorithm (Rosenbrock's method) is the currently used NP algorithm. This was chosen because it could not be assumed that either first or second derivatives of the criterion function would be available. In particular, the derivatives would not be available if the detection contours were drawn as a Sketch Model by the operator.

The problems used in the experiment recently conducted did not contain Sketch Models. Instead the detection contours were drawn from stored analytical functions which do have derivatives. Gradient search is usually faster than the gradient-free approach. Thus it would be worthwhile to implement a gradient search and test its performance against that already obtained using Rosenbrock's method (the variable metric method is recommended). Further, a gradient search could be used even in those cases where analytic derivatives are difficult to compute. A difference approximation would then be used for the derivatives.

REFERENCES

1. Irving, G.W., et al., Experimental Investigation of Sketch Model Accuracy and Usefulness in a Simulated Tactical Decision Aiding Task. Report No. 215-3, Integrated Sciences Corporation, Santa Monica, CA, May 1977.

2. Payne, J.R. and Rowney, J.V., ONRODA Warfare Scenario. SRI/NWRC-RM-83, Stanford Research Institute, Menlo Park, CA, June 1975. AD# AO 16625.

3. Rosenbrock, H.H., "An Automatic Method for Finding the Greatest or the Least Value of a Function." The Computer Journal, Vol.3, 1960, pp. 175-184.

4. Powell, M.J.D., "An Efficient Method of Finding the Minimum of a Function of Several Variables Without Calculating Derivatives." The Computer Journal, Vol. 7, 1964, pp. 155-162.

5. Winer, B.J. Statistical Principles in Experimental Design, 2nd edition, New York, McGraw Hill, 1971.

6. Zehna, Park W. (Ed.), Selected Methods and Models in Military Operations Research, Department of Operations Research and Administrative Sciences, Navy Postgraduate School, Monterey, CA, 1971.

7. Nemhauser, G.L. Introduction to Dynamic Programming, John Wiley and Sons, New York, 1967.

APPENDIX A

CONTOUR DRAWING ALGORITHM

APPENDIX A: CONTOUR DRAWING ALGORITHM

For the experiment, it was necessary to come up with a set of "true" composite detection rate contours for each configuration of sensor locations and sensor sizes used. Each set of true contours corresponds to a "Sketch Model" the subjects were required to draw, and the true contours were displayed as feedback to the subjects at the end of each trial under System Concept B. The development of the contour following algorithm which traces out the contours required is described below.

A three-dimensional function f(x,y) defined in a region $x_o \le x \le x_t$, $y_o \le y \le y_t$ can be represented in two-dimensional space as a series of iso-altitude contours. In other words, the curve $f(x,y) = h$ can be plotted for every altitude h we have chosen. Each contour thus represents points at the same specified altitude.

The contour following algorithm requires two major steps. In the first step, points at each desired altitude level are sampled and stored in a list. Then each list of points is examined and the points connected so that the resulting curves describe the true contours reasonably well. These two steps are further explained below.

To simplify the exposition, let us for the moment concern ourselves with one altitude level only. The method is easily extended to more than one altitude.

Suppose we want to find the contours for an altitude h. First we search in the x-y domain for points at this altitude. We try to find a sufficient number of points (x,y) such that $f(x,y) = h$. This is done in two passes.

In the first pass, a flowchart of which is given in Figure A1, we lay down grid lines δ units apart in the Y direction. The value δ is selected such that the grid lines are close enough for the resulting contour

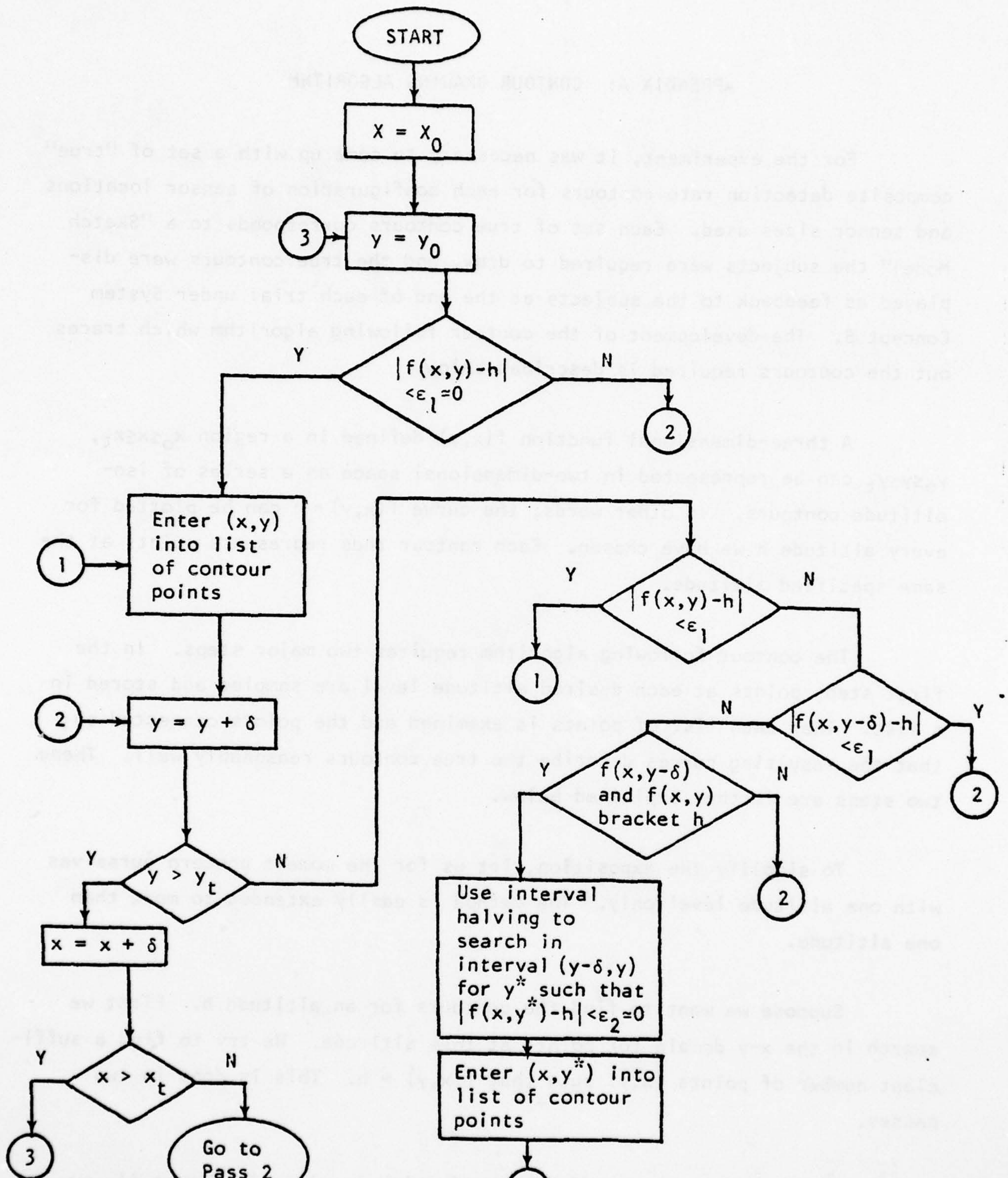

START
$X = X_0$
$y = y_0$
$|f(x,y)-h| < \varepsilon_1 \approx 0$
Y
N
Enter (x,y) into list of contour points
$y = y + \delta$
$y > y_t$
$x = x + \delta$
$x > x_t$
Go to Pass 2
$|f(x,y)-h| < \varepsilon_1$
$|f(x,y-\delta)-h| < \varepsilon_1$
$f(x,y-\delta)$ and $f(x,y)$ bracket h
Use interval halving to search in interval $(y-\delta,y)$ for y^* such that $|f(x,y^*)-h| < \varepsilon_2 \approx 0$
Enter (x,y^*) into list of contour points
1
2
3

Figure A1. Flowchart for First Step of Contour Following Algorithms, Pass 1.

points not to be too sparse. At the same time the grid lines are not so close that excessive time is spent in the computations. Each grid line is then divided into intervals of width δ (see Figure A2). This leads to another consideration for the value of δ in that the finer the grid resolution the easier it is to detect sharp ridges and peaks. The functional values at the end points of each interval are checked to see if the desired altitude h is bracketed. In other words, we check to see if h is between the values of the function at the two end points. If bracketing occurs we know we can find a point within the interval where the value of f is sufficiently close to h. We do this by successively halving the interval until we are close enough to the desired point. Each point on the contour found is put into a list for later use.

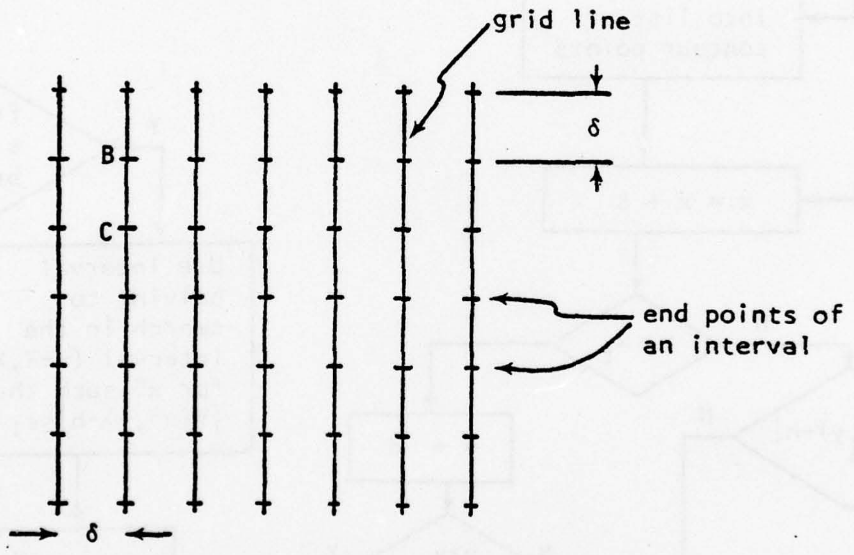


Figure A2. Example of Grid Line Division. (Each grid line is divided into intervals, such as BC.)

At the completion of the first pass, we move to the second pass; the corresponding flowchart is shown in Figure A3. Here we lay down a second grid of the same grid size δ, but this time the grid lines run in the X direction. Each grid line is divided into intervals, each of length δ. This second grid is necessary because the first grid cannot "see" very well those contours that are parallel to the grid lines, as shown in Figure A4.

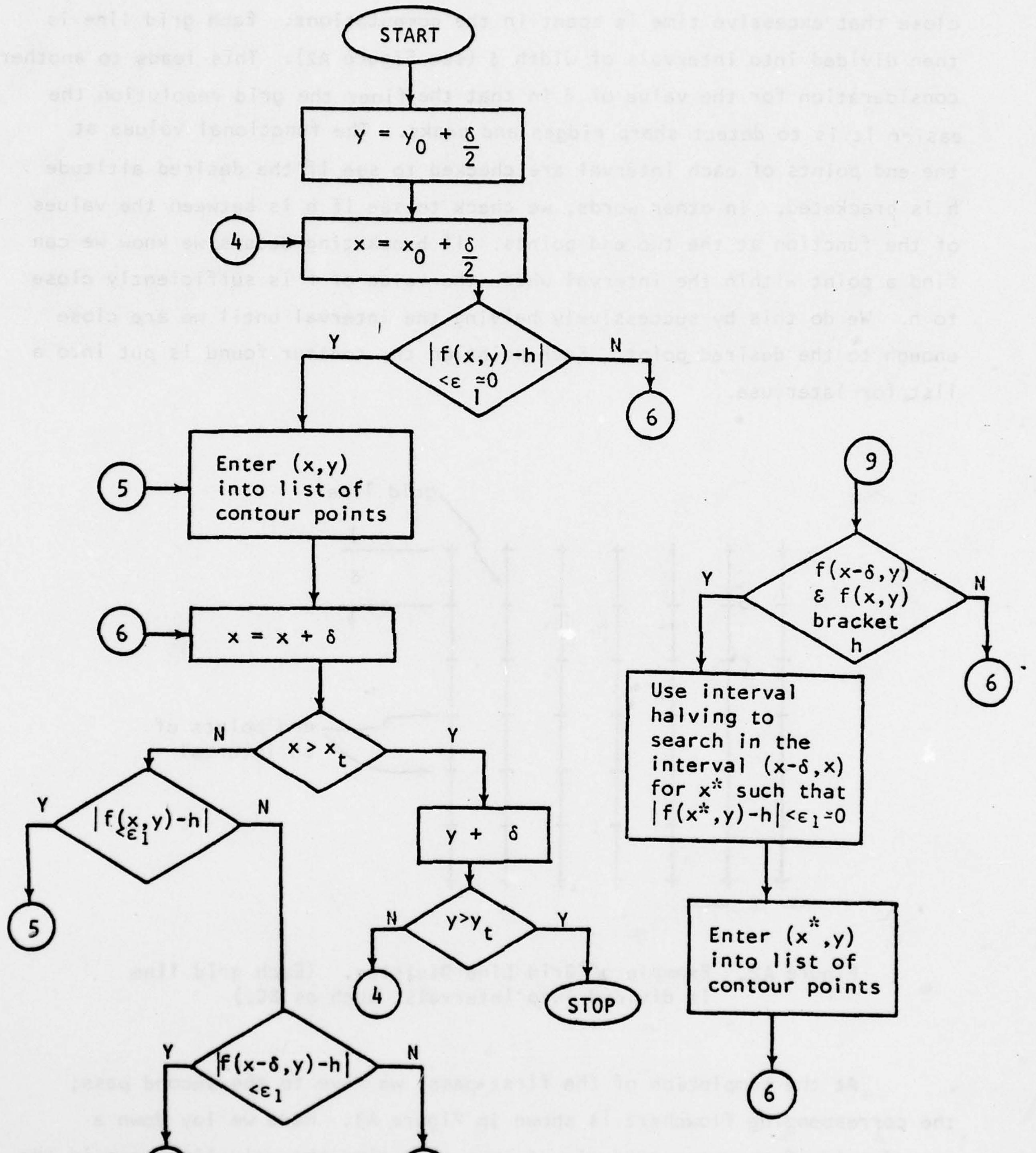

START
$y = y_0 + \frac{\delta}{2}$
4
$x = x_0 + \frac{\delta}{2}$
$|f(x,y)-h| < \varepsilon_1 \simeq 0$
Y
N
6
5
Enter (x,y) into list of contour points
6
$x = x + \delta$
$x > x_t$
N
Y
$|f(x,y)-h| < \varepsilon_1$
Y
N
5
$y + \delta$
$y > y_t$
N
Y
4
STOP
$|f(x-\delta,y)-h| < \varepsilon_1$
Y
N
6
9
9
$f(x-\delta,y)$ & $f(x,y)$ bracket h
Y
N
6
Use interval halving to search in the interval $(x-\delta,x)$ for x^* such that $|f(x^*,y)-h| < \varepsilon_1 \simeq 0$
Enter (x^*,y) into list of contour points
6

Figure A3. Flowchart for First Step of Contour Following Algorithm, Pass 2.

Since the second grid is perpendicular to the first, it can be used to remedy problems such as that depicted in Figure A4. The second grid is arbitrarily displaced from the first so that grid points from the two will not overlap, but this is not necessary.

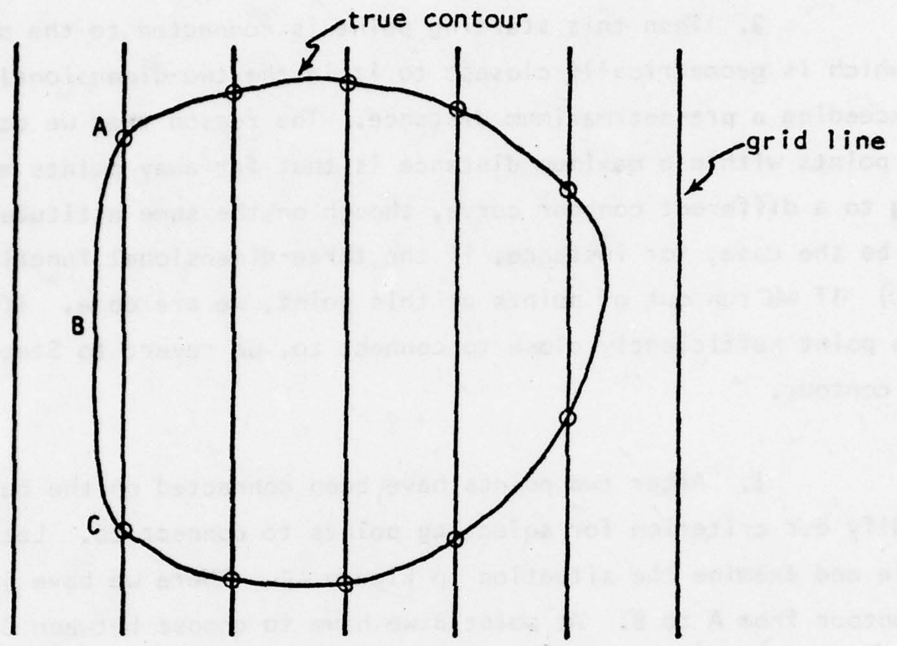


Figure A4. Error When Using a Single Grid. (With grid lines in the shown direction, the ABC portion of true contour is not detected by the grid lines. The dots are contour points picked up by grid.)

With the second grid, each interval on a grid line is checked for bracketing of the desired altitude h. Each occurrence signifies a contour point within the interval under consideration. Searching by interval halving yields a point on the contour, and this point is added to our list of contour points.

At the completion of the second pass we have a list of points at the altitude h. Obviously an empty list means that no contour at that altitude could be found. The next step is to reconstruct contours with this non-empty list of points.

We generate the contours by connecting the points in a meaningful manner. A simplified description of the algorithm used is given below.

1. To begin, a point is obtained from the previously generated list as a starting point for a new contour.

2. Then this starting point is connected to the point in the list which is geometrically closest to it in the two-dimensional plane, but not exceeding a pre-set maximum distance. The reason that we consider only those points within a maximum distance is that far away points most likely belong to a different contour curve, though on the same altitude. (This would be the case, for instance, if the three-dimensional function is bi-modal.) If we run out of points at this point, we are done. If we fail to find a point sufficiently close to connect to, we revert to Step 1 to start a new contour.

3. After two points have been connected on the current contour, we modify our criterion for selecting points to connect to. Let us pause for a while and examine the situation in Figure A5. There we have just connected the contour from A to B. At point B we have to choose between C and D as the next point to connect to. Point D is closer than point C, but going from B to D involves a greater directional change, as measured by $\Delta\Theta_2$ in the figure. Assuming our contour to be a smooth curve with minimal zig-zagging, we may want to connect to point C instead. In order to accomplish this aim, the criterion function for selecting the next point is modified to be

$$\Delta S + \beta \cdot \Delta\Theta_1$$

where

ΔS is distance to candidate point (e.g., the length BC)

$\Delta\Theta$ is the directional change (e.g., $\Delta\Theta_1$)

β is the weighting factor to be selected by trial and error to yield acceptable results for types of curves under consideration

With this modified "distance" criterion, we always select a point with the smallest criterion function. As before, only those points which lie within a sufficiently small distance will be considered. Before connecting to the point selected, it is compared with the starting point (obtained in Step 1), if the starting point is close enough. Between the two, the one with the smaller criterion function is the next point to connect to. This provision allows us to close the contour curve when we reapproach the starting point.

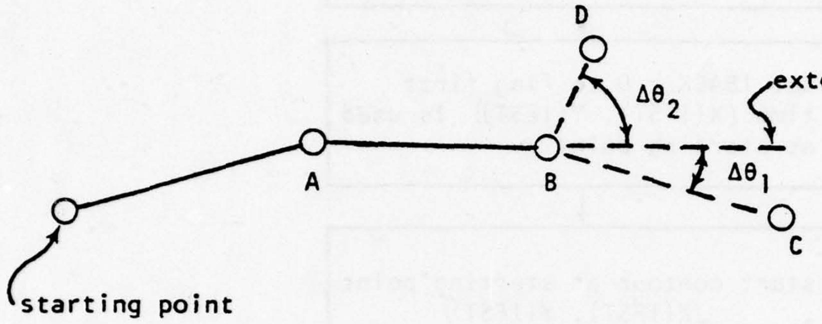


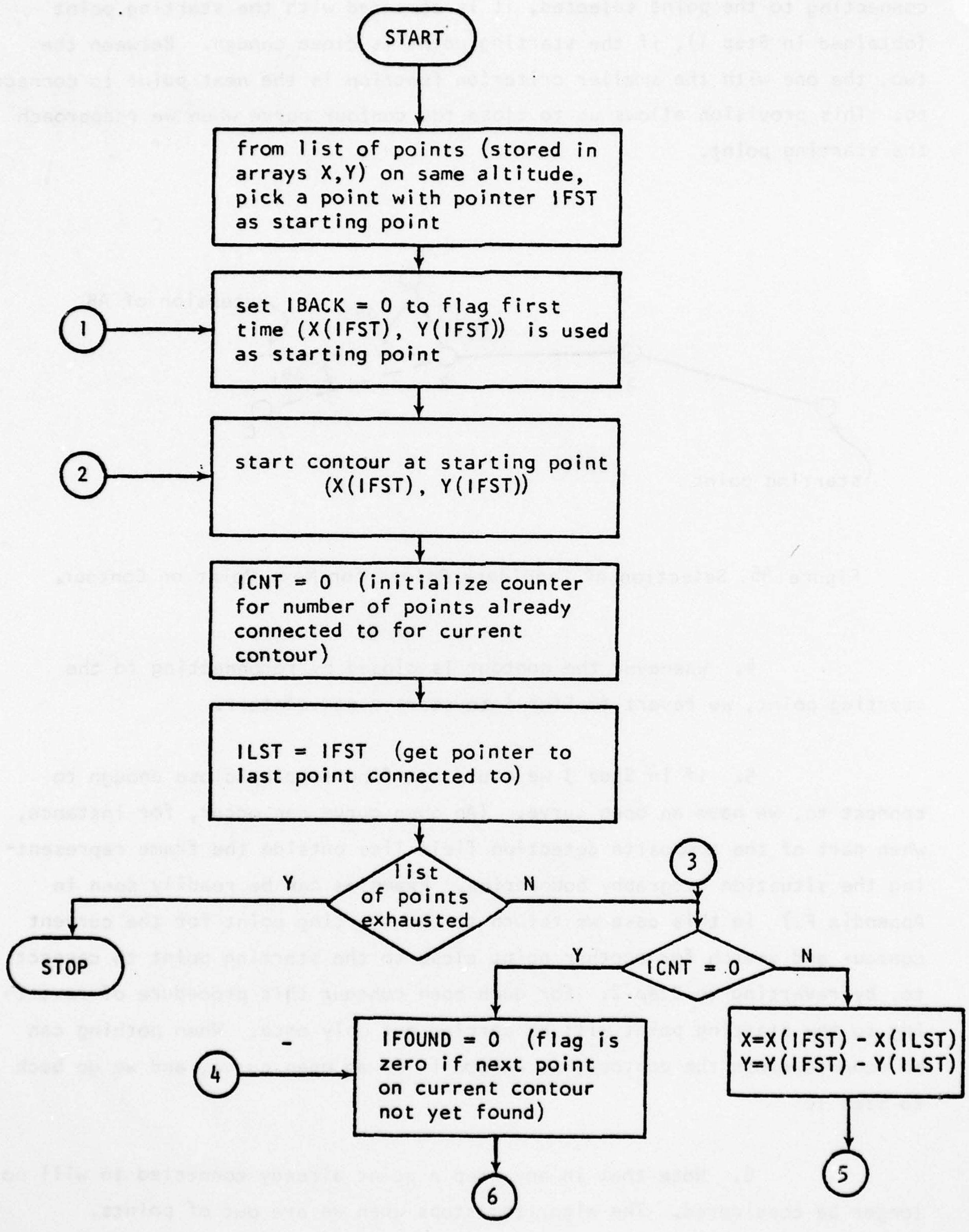
Figure A5. Selection of Candidate Points for Next Point on Contour.

4. Whenever the contour is closed by reconnecting to the starting point, we revert to Step 1 to start a new contour.

5. If in Step 3 we could not find a point close enough to connect to, we have an open curve. (An open curve can occur, for instance, when part of the composite detection field lies outside the frame representing the situation geography boundaries. Examples can be readily seen in Appendix F.) In this case we return to the starting point for the current contour and search for another point close to the starting point to connect to, by reverting to Step 2. For each open contour this procedure of reverting to the starting point will be carried out only once. When nothing can be done to close the contour, we assume it is an open curve, and we go back to Step 1.

6. Note that in any step a point already connected to will no longer be considered. The algorithm stops when we are out of points.

Figure A6 provides a more detailed representation of the algorithm.



START
from list of points (stored in arrays X,Y) on same altitude, pick a point with pointer IFST as starting point
1
set IBACK = 0 to flag first time (X(IFST), Y(IFST)) is used as starting point
2
start contour at starting point (X(IFST), Y(IFST))
ICNT = 0 (initialize counter for number of points already connected to for current contour)
ILST = IFST (get pointer to last point connected to)
list of points exhausted
Y
N
3
STOP
ICNT = 0
Y
N
4
IFOUND = 0 (flag is zero if next point on current contour not yet found)
X=X(IFST) - X(ILST)
Y=Y(IFST) - Y(ILST)
6
5

Figure A6. Flowchart for Contour Following Algorithm, Second Step.

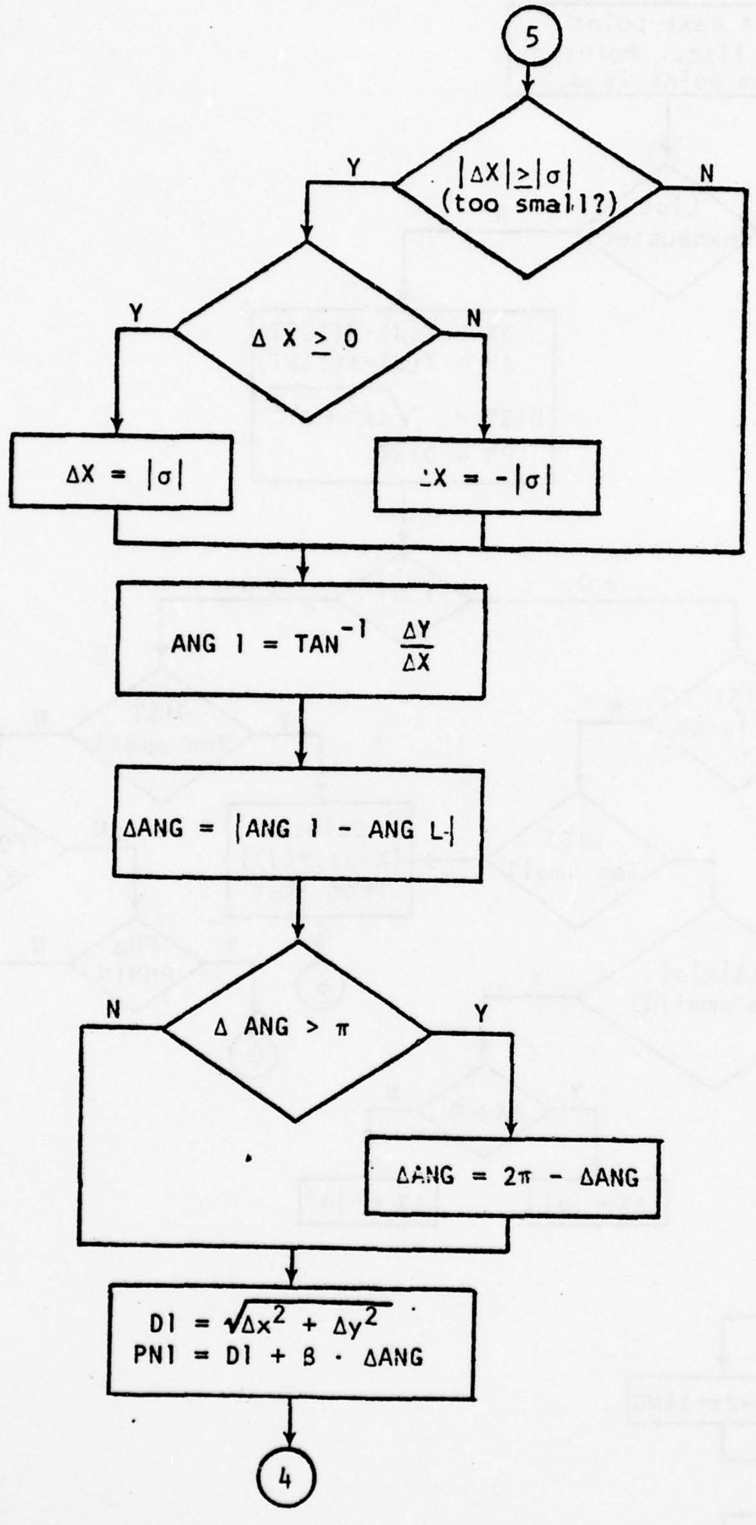


Figure A6. (Continued).

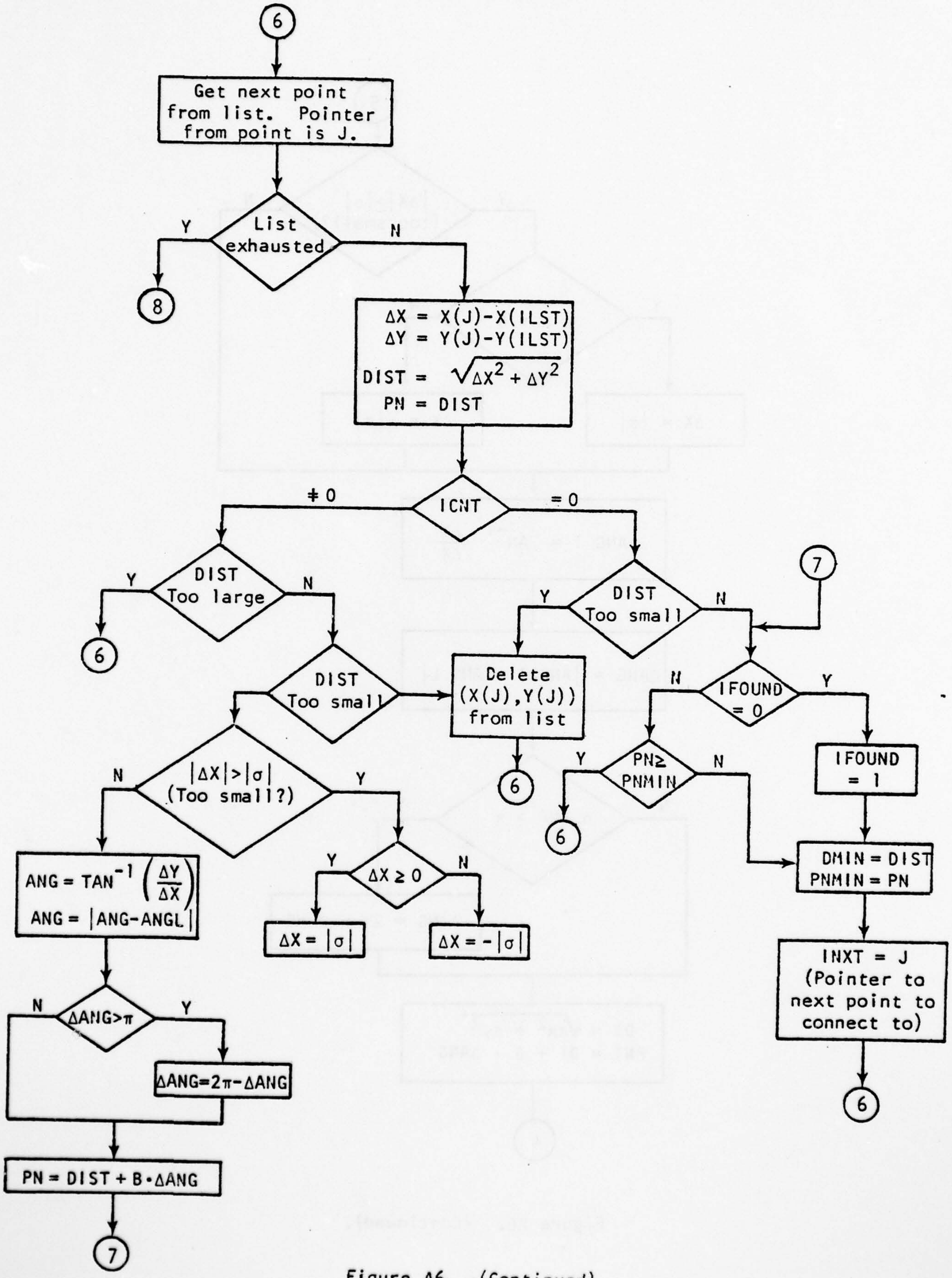


Figure A6. (Continued).

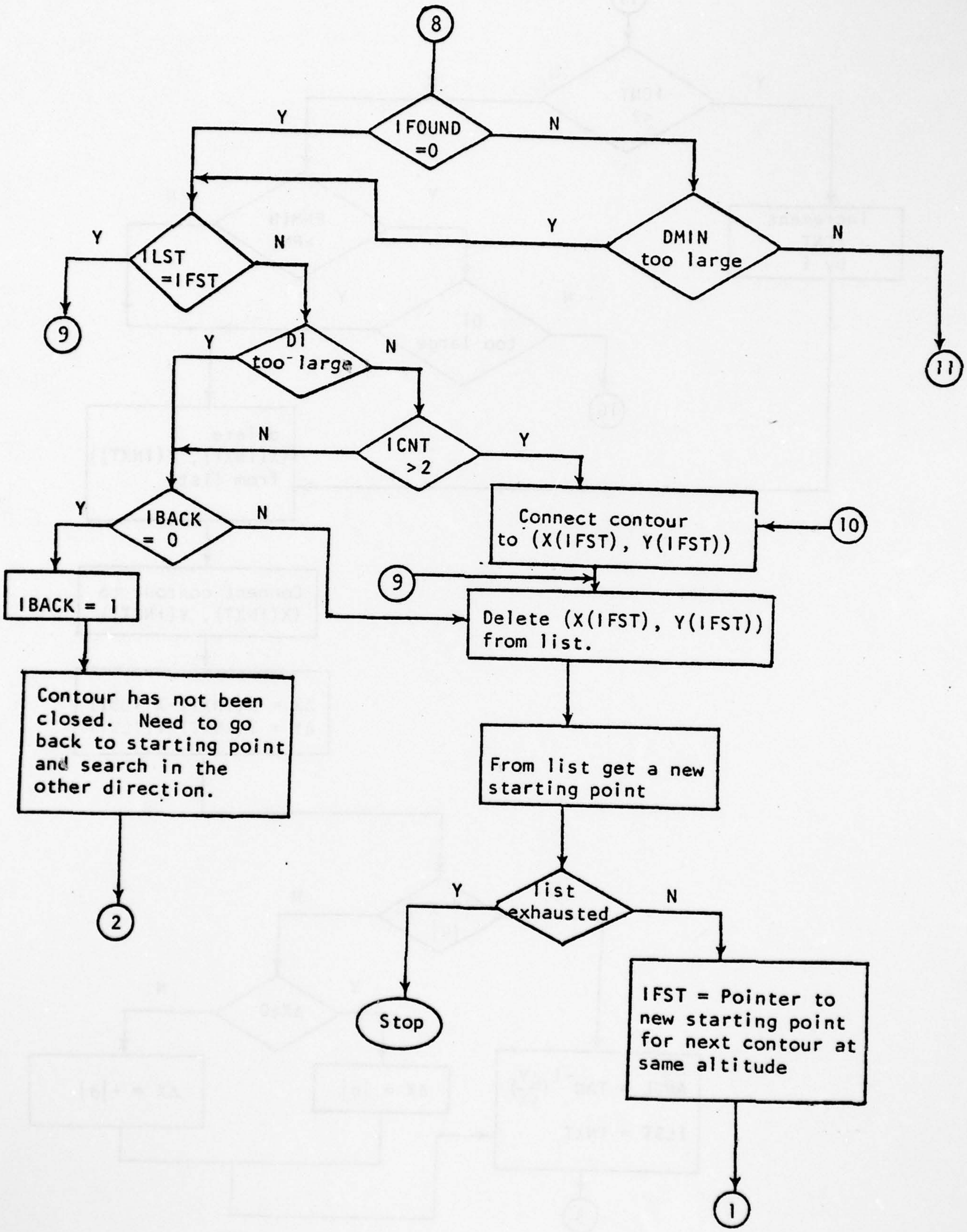
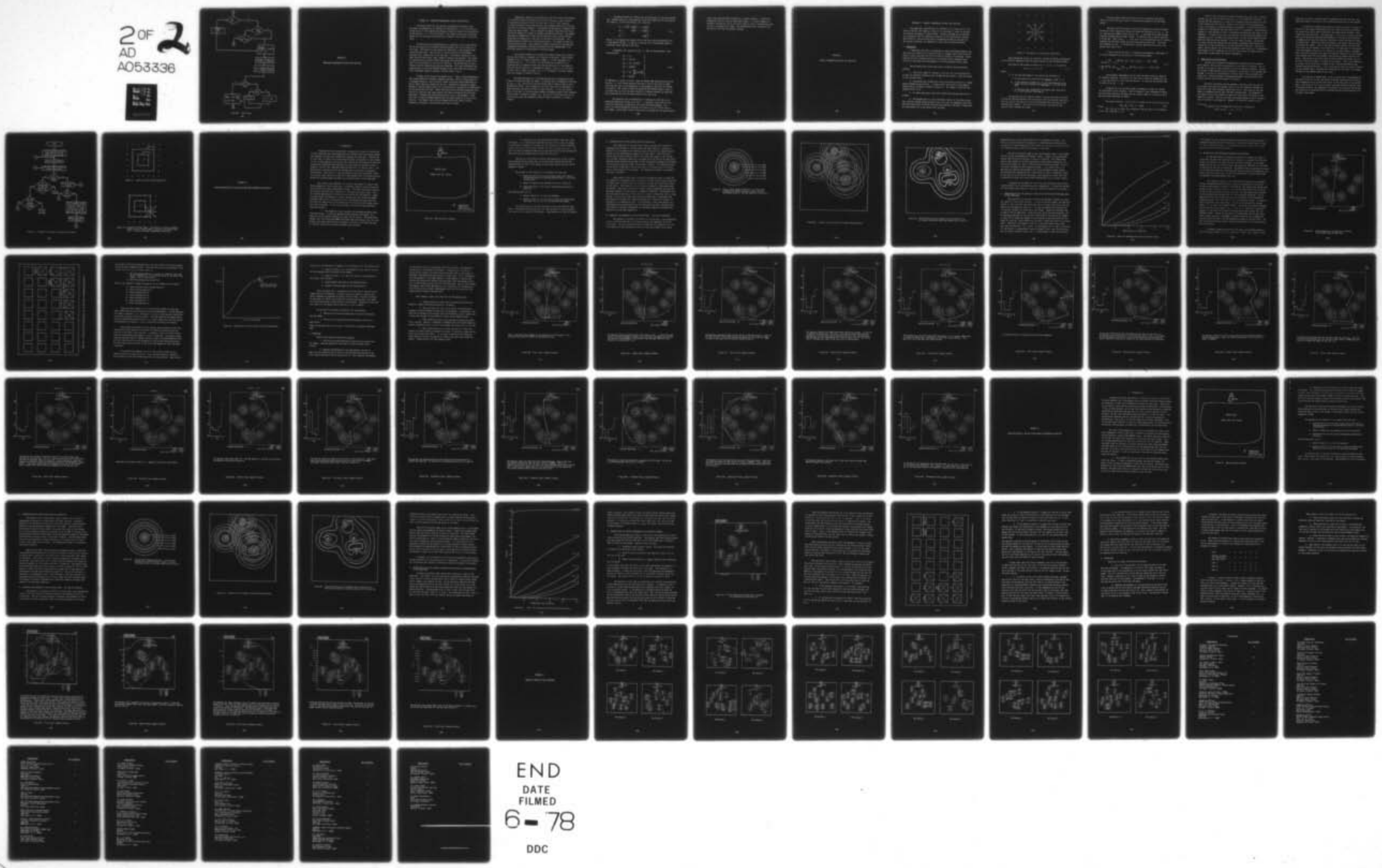

8
IFOUND =0
Y
N
DMIN too large
Y
N
11
ILST =IFST
Y
N
9
D1 too large
Y
N
ICNT >2
N
Y
IBACK = 0
Y
N
IBACK = 1
Connect contour to (X(IFST), Y(IFST))
10
9
Delete (X(IFST), Y(IFST)) from list.
Contour has not been closed. Need to go back to starting point and search in the other direction.
2
From list get a new starting point
list exhausted
Y
N
Stop
IFST = Pointer to new starting point for next contour at same altitude
1

Figure A6. (Continued).

AD-A053 336 INTEGRATED SCIENCES CORP SANTA MONICA CALIF F/G 5/8
EXPERIMENTAL INVESTIGATION OF THE USEFULNESS OF OPERATOR AIDED --ETC(U)
JAN 78 D H WALSH, M D SCHECHTERMAN N00014-75-C-0811
UNCLASSIFIED ISC-215-4 NL

2 OF 2
AD
A053336



END
DATE
FILMED
6-78
DDC

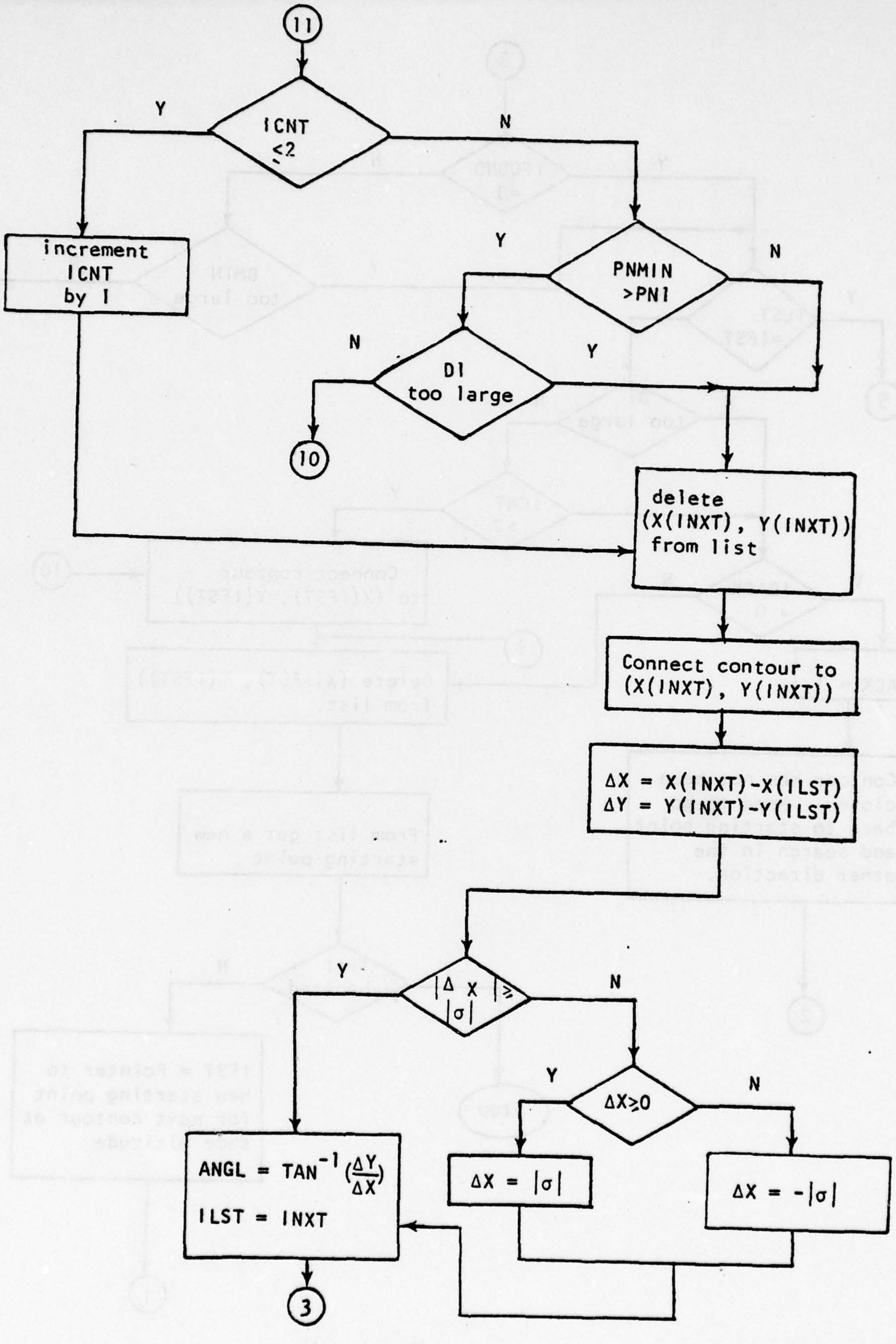


Figure A6. (Continued).

APPENDIX B

NONLINEAR PROGRAMMING OPTIMAL PATH SOLUTION

APPENDIX B: NONLINEAR PROGRAMMING OPTIMAL PATH SOLUTION

The method chosen for the nonlinear programming optimization was developed and proposed by H.H. Rosenbrock (Reference 3). It is an attractive method to use for this application because it does not require the calculation of derivatives. It is fairly efficient in the number of function evaluations needed while at the same time being able to handle a wide variety of function types.

The derivative-free characteristic is necessary in this application due to the nature of the function being optimized, the utility function in this case. The function is not explicitly expressed in the variables that are being controlled. The utility is a direct function of performance measures (e.g., penetration ratio) and cost. The control variables on the other hand include sensor types, number of sensors and location. Once these are specified the simulator (NIBS) calculates the performance figures to be used in the evaluation of this utility function. The derivatives of such a function clearly cannot be analytically obtained. This fact eliminated from consideration all the optimization methods which require derivative calculations (conjugate gradient, Newtons, Fletcher-Powell, etc.).

A number of derivative-free methods exist. These include Rosenbrock's, the simplex method of Himsworth, Spendley and Hext, Smith's method based on a conjugate direction, and of course, simple univariate search (Reference 4). Any of these might be suitable for the job. Rosenbrock's method was chosen because it had the added flexibility of allowing the introduction of constraints on the controlled variables. These functions could be defined separately from the utility function that would control the region through which search was permitted. Although constrained optimization is not a requirement, it was felt that it might be needed depending on the global behavior of the utility function. This added flexibility was deemed sufficient cause to select Rosenbrock's method as the candidate optimization scheme.

Rosenbrock's method is an extension of the univariate search method. In univariate search the minimum of a function $u(x_1, x_2, \ldots, x_n)$ is found by searching along each of the x_i directions in turn. After reducing u as far as possible with each variable, the procedure moves on to the next variable in a cyclical fashion. This method can bog down on elongated functions with deep troughs. This is because the search directions are fixed and do not change as a result of progress through the function. The method developed by Rosenbrock is meant to eliminate this fault without adding a great deal of complexity or the need for derivative calculations. The method basically consists of finding two factors: (1) Length of Step and (2) Direction of Step. Using these two factors according to the algorithm proposed by Rosenbrock, function minimization can be accomplished.in an efficient manner on a wide variety of function types.

The simplest problem is to decide the length of step to be taken in the desired direction, assuming this direction to be known. The principle adopted is to try a step of arbitrary length e. If this succeeds, e is multiplied by $\alpha > 1$. If it fails, e is multiplied by $-\beta$ where $0 < \beta < 1$. "Success" here is defined to mean that the new value of u is less than or equal to the old value for a minimization problem. Thus if e is initially so small that it makes no change in u, it is increased on the next attempt. Each such attempt is called a "trial."

The remaining factor is to decide when and how to change the directions $\underline{\xi}$ in which the steps are taken. The method uses n orthogonal directions $\xi_1, \xi_2, \ldots, \xi_n$. One trial of the univariate type is made in each of the n directions in turn. This is done until at least one trial is successful in each direction, and one has failed in each direction. It will be noticed that a trial must in the end succeed because e becomes so small after repeated failures that it causes no change in u. The set of trials made with one set of directions, and the subsequent change of these directions, is called a "stage."

The method chosen for finding the new directions of ξ was the following. Suppose that d_1 is the algebraic sum of the lengths of all the successful steps e_1, in the direction ξ_1, etc. Then let

$$\left.\begin{aligned} A_1 &= d_1\xi_1^0 + d_2\xi_2^0 + \dots + d_n\xi_n^0 \\ A_2 &= \qquad\quad d_2\xi_2^0 + \dots + d_n\xi_n^0 \\ &\dots\dots \\ A_n &= \qquad\qquad\qquad\qquad d_n\xi_n^0 \end{aligned}\right\} \qquad (B.1)$$

Thus A_1 is the vector joining the initial and final points obtained by use of the vectors $\xi_1^0, \xi_2^0, \dots, \xi_n^0$, A_2 is the sum of all the advances made in directions other than the first, etc.

Orthogonal unit vectors $\xi_1^1, \xi_2^1, \dots, \xi_n^1$, are now obtained in the following way:

$$\left.\begin{aligned} B_1 &= A_1 \\ \xi_1^1 &= B_1/|B_1| \\ B_2 &= A_2 - A_2\cdot\xi_1^1\xi_1^1 \\ \xi_2^1 &= B_2/|B_2| \\ &\dots\dots \\ B_n &= A_n - \sum_{j=1}^{n-1} A_n\cdot\xi_j^1\xi_j^1 \\ \xi_n^1 &= B_n/|B_n \end{aligned}\right\} \qquad (B.2)$$

No ambiguity is likely to arise, since the method used ensures that no d can be zero. It is of course possible that one or more of the d are so small that they are lost in the summations of equations (B.1), but this is unlikely in practice. The result of applying equations (B.1) and (B.2) several times is to ensure that ξ_1 lies along the direction of fastest advance, ξ_2 along the best direction which can be found normal to ξ_1, and so on.

The numerical work of developing this process was carried out to determine appropriate values for α and β. In addition, tests were run on a variety of functions in comparison with other available methods. As a result of testing Rosenbrock selected the values $\alpha = 3$, $\beta = 0.5$ for use in his method. Using these values he found that his method was not significantly slower than the available alternatives in simple problems. In difficult problems he claims it may be a good deal faster. It is well adapted to automatic calculation, and is not easily upset by minor irregularities such as occur in asymmetrical ridges. The method permits the introduction of constraints into the minimization problem.

APPENDIX C

DYNAMIC PROGRAMMING OPTIMAL PATH SOLUTION

APPENDIX C: DYNAMIC PROGRAMMING OPTIMAL PATH SOLUTION

The experiment required best (or "answer") air strike paths (in the sense of optimal with respect to the utility function) in order to evaluate the operator estimates of optimal paths. Dynamic programming was selected as a convenient tool for obtaining solutions to this optimization problem. Specifically, a grid-oriented dynamic programming method was selected for the optimal solutions; its adaptation to the study is explained below.

1. Background

A detection rate field is generated by sensor performance models, and the task is to transit this field arriving at a target from a given starting point in such a way that the utility criterion function is maximized. Recall that the utility criterion function is a function of fuel remaining upon arrival at the target and the probability of being detected along a path.

The following restrictions were used to simplify the optimization problem.

1. The entire region of interest is put on a 16 x 16 quadruled grid so that all possible trajectories go through the grid points. Each grid point is also called a node.

2. Transitions are allowed only from a grid point to one of its immediately neighboring grid points. In other words, only cardinal and diagonal transitions are allowed, as shown in Figure C1. The "legal" transitions are numbered from 1 to 8.

3. For each transition, only one of three pre-selected velocities is allowed.

4. The problem then is, given two nodes A and B, to find both a path that transits through the grid points from A to B and its component velocities such that the utility criterion function associated with the path is maximized. For convenience call B the target and A the base.

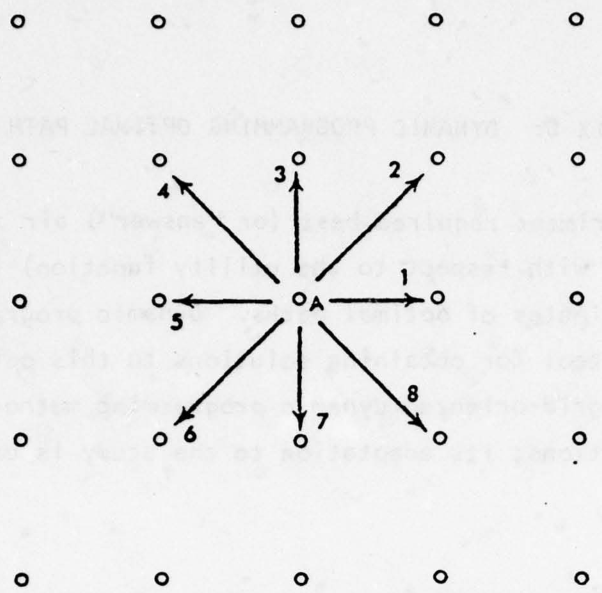


Figure C1. Allowable Air Strike Path Transitions.

This problem can be put in a form for solution by dynamic programming. Let us define a sequential decision making process by the following system:

The state of the system at stage k (k = 1, 2, 3, ...) is given by

$$X_k = (i, j, v, P_c, F) \tag{C.1}$$

where

(i, j) are the coordinates of the node we are currently at

v is the velocity for the transition <u>ending</u> at node (i, j)

P_c is the cumulative probability of not being detected by the sensor field for optimal path <u>starting</u> at i, j) and ending at B

F is the total fuel consumption for optimal path starting at (i, j) and ending at the target B

The decision D_k to be made at stage k is one of the eight allowable transitions defined previously and the velocity for the selected transition. Let $r(X_k, D_k)$ be the return (or utility) associated with the state X_k and decision D_k at stage k. Also let g be the criterion function for the returns from a sequence of stages.

Since an optimal path going from A to B is an optimal path going from B to A and vice versa, we will formulate the problem in the following manner:

Starting with the system at state X_1 corresponding to node B (target) we want to find a sequence of decisions (paths and velocities $D_1, D_2, \ldots, D_N$ for some positive number N such that $D_1, D_2, \ldots, D_N$ takes the system to some state X_N corresponding to node A (base). In addition, the criterion function $g\,[r(X_N, D_N), r(X_{N-1}, D_{N-1}), \ldots, r(X_1, D_1)]$ is to be maximized over all possible sequences of decisions (paths and velocities) going from B to A. This decision sequence $D_1, D_2, \ldots, D_N$ defines an optimal path from B to A and is an optimal path from A to B.

If the criterion function g is separable and monotonic, (Reference 7), it can be decomposed into g_1 and g_2 in the following manner.

$$\max_{D_k, D_{k-1}, \ldots, D_1} \left\{ g[r(X_k, D_k), r(X_{k-1}, D_{k-1}), \ldots, r(X_1, D_1)] \right\} \tag{C.2}$$

$$= \max_{D_k} g_1[r(X_k, D_k), \max_{D_{k-1}, \ldots, D_1} g_2(r(X_{k-1}, D_{k-1}), \ldots, r(X_1, D_1))]$$

Using dynamic programming, we can solve the above recursive equation by backward recursion. We start at X_1 and select a decision sequence $D_1, D_2, \ldots, D_N$, one stage at a time, that maximizes g. At state X_N the optimal path is given by $D_1, D_2, \ldots, D_N$.

We would carry out as many stages as necessary in order to include all grid points in our solution. What this means is that regardless of which point we later choose as the base, we can retrieve from our solution an optimal path going from base to target.

The return function r in Eq. (C.2) is related to the utility function by

$$r(X_m, D_m) = U(F_m, (1 - Pc_m)) \tag{C.3}$$

where

$X_m = (i_m, j_m, r_m, Pc_m, F_m)$ as defined in Eq. (C.1) and U is as defined in Eq. (1) in Section II.C.2.

Since in Eq. (C.2) we are trying to maximize $r(X_k, D_k) = U(F_k, (1-P_{ck}))$ at node (i_k, j_k), the criterion function g is a 1st projection function.* It follows that the function g_1 is also a 1st projection function and we can arbitrarily define the function g_2. However, the validity of the decomposition in Eq. (C.2) hinges on g being separable and monotonic. The criterion function g being a 1st projection function implies that it is separable. Unfortunately, though it is not obvious at first, a few sample calculations show that g is not monotonic. In spite of this fact, by using Eq. (C.2) and applying dynamic programming, we still arrived at an optimal answer in most of the cases we tried. When the solution was not optimal, it was very nearly optimal. We decided to retain the utility function and the dynamic programming optimization procedure due to time limitations to get the experiment under way. The experimental results support our belief that meaningful data could still be obtained, even though strictly speaking, our utility function may not (under certain conditions) be optimizable by dynamic programming.

2. <u>Description of the Algorithm</u>

We have just discussed the mathematical description of our optimal path problem. Now we desire a procedure for arriving at a solution. Let us go back to the grid we have laid over the entire region of interest. Picture our target B (destination) as being located on a grid point. Our problem then becomes, if we pick a starting point A (base), how to find an optimal path going to B. Using the dynamic programming approach, we would solve for an optimal path for each starting point on the grid.

Basically the method involves backward recursion, where we start at the target B and solve for optimal paths for all the immediate neighboring points of B that have allowable transitions into B. Then we go to the next layer of neighboring points and find their optimal paths. We propagate in this manner by going to successive layers of points until all points in the region are covered. This is a multiple pass method because we will reiterate until a solution is converged to. However, for practical reasons we will

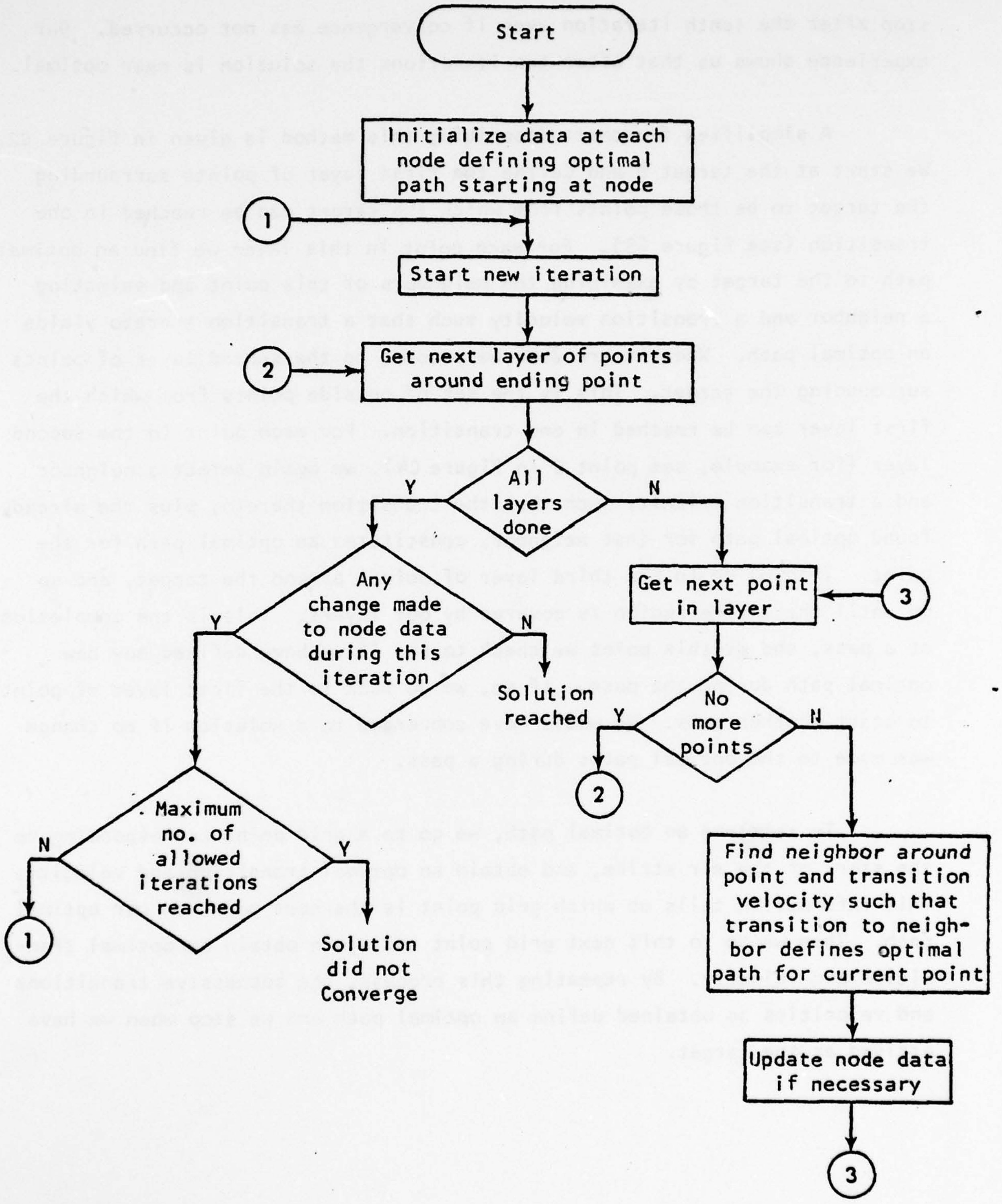
*In general, the ith projection function f_i is defined as

$$f_i(y_1, y_2, y_3, \dots y_i, \dots) = y_i$$

stop after the tenth iteration even if convergence has not occurred. Our experience shows us that after ten iterations the solution is near optimal.

A simplified flowchart describing this method is given in Figure C2. We start at the target B and define the first layer of points surrounding the target to be those points from which the target can be reached in one transition (see Figure C3). For each point in this layer we find an optimal path to the target by examining the neighbors of this point and selecting a neighbor and a transition velocity such that a transition thereto yields an optimal path. When we are done we proceed to the second layer of points surrounding the target. This is the set of outside points from which the first layer can be reached in one transition. For each point in the second layer (for example, see point C in Figure C4), we again select a neighbor and a transition velocity such that the transition thereto, plus the already found optimal path for that neighbor, constitutes an optimal path for the point. Then we go to the third layer of points around the target, and so on until the entire region is covered by our layers. This is the completion of a pass, and at this point we check to see if we have defined any new optimal path during the pass. If so, we go back to the first layer of points to start another pass. We would have converged to a solution if no change was made to the optimal paths during a pass.

To retrieve an optimal path, we go to a grid point corresponding to the start of the air strike, and obtain an optimal transition and velocity. This information tells us which grid point is the next point on our optimal path. Then we go to this next grid point and again obtain an optimal transition and velocity. By repeating this process, the successive transitions and velocities so obtained define an optimal path and we stop when we have arrived at the target.



Start
Initialize data at each node defining optimal path starting at node
1
Start new iteration
2
Get next layer of points around ending point
All layers done
Y
N
Any change made to node data during this iteration
Y
N
Solution reached
Get next point in layer
3
No more points
Y
N
2
Maximum no. of allowed iterations reached
N
Y
1
Solution did not Converge
Find neighbor around point and transition velocity such that transition to neighbor defines optimal path for current point
Update node data if necessary
3

Figure C2. Flowchart for Dynamic Programming Procedure.

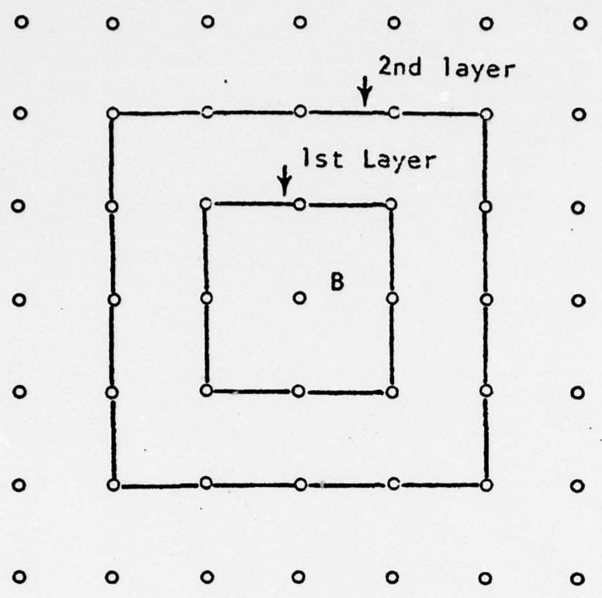


Figure C3. Layers of Points Around Target (B).

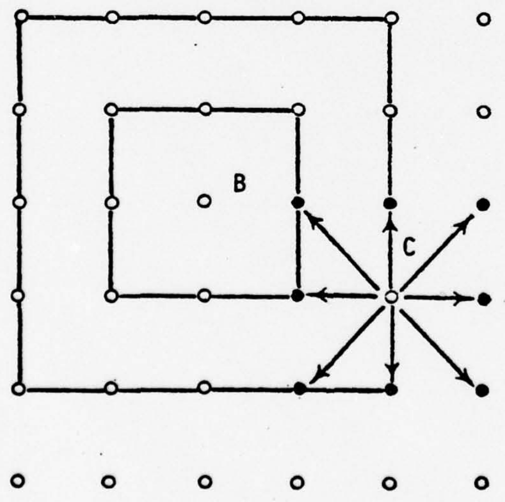


Figure C4. Potential Optimal Nodes. (From point C, we are to decide which of the filled-in nodes is to be on its optimal path to B. An arrow indicates a candidate transition.)

APPENDIX D

TRAINING MATERIALS FOR OAO USING NONLINEAR PROGRAMMING ALGORITHM

I. INTRODUCTION

Integrated Sciences Corporation is conducting a study for the Office of Naval Research that investigates ways to allocate functions between humans and computers so that their respective strengths are best used. The portion of the study in which you are participating seeks to determine to what extent, if any, a human operator can aid and thus improve the performance of selected optimization techniques when applied to a naval tactical decision aiding problem. We call this "operator-aided optimization," or OAO for short. The optimization technique you will be working with is nonlinear programming optimization. Don't worry if you are unfamiliar with this technique. Even if you have never heard of it, you will learn enough about its characteristics during the training phase to enable you to perform well on the experiment.

Your role in the experiment is to act as the member of a Naval Task Force Commander's (TFC's) staff who is planning a tactical airstrike against the airfield on a place called ONRODA Island. Your Naval Task Force consists of aircraft carriers, their squadrons of aircraft, and escort ships. They are located approximately at the point marked with an X in Figure D1. About ten enemy ships are located in a region between your Task Force and ONRODA. Important parts of air strike planning are (a) deciding the path that the aircraft will take to get to the target and (b) strike aircraft speeds along the legs of the path. As air strike planner, you must be concerned about these two factors:

1. The probability that aircraft will be detected before they reach the target. If they are detected before reaching the target, the enemy will be at maximum readiness to repel the air strike. The enemy ships between your Task Force and ONRODA have radar that could detect your aircraft. However, the enemy ships have no interceptor aircraft nor do they have guns or missiles that would be effective against your aircraft.

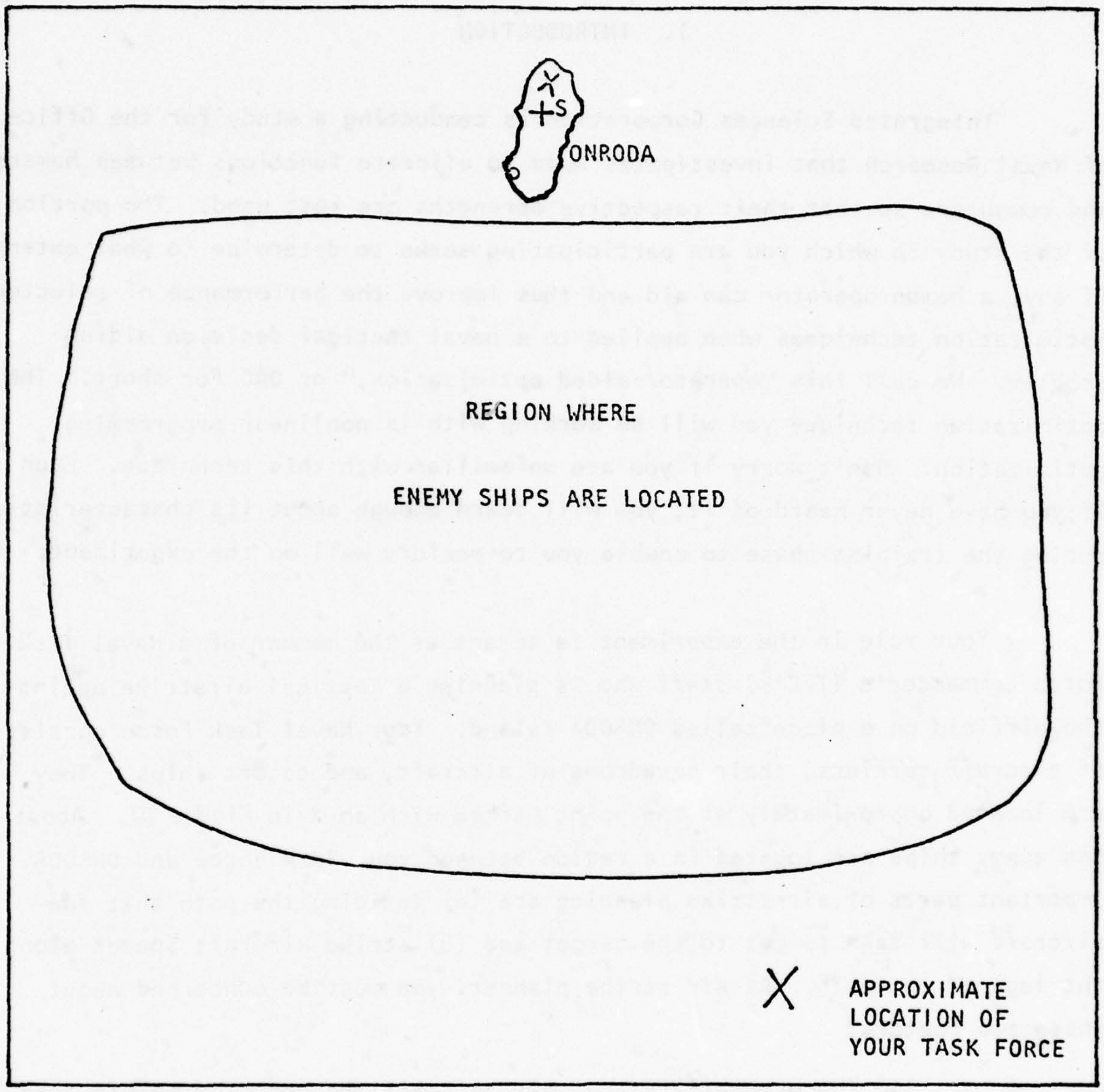


Figure D1. Map of Area of Interest.

2. Amount of fuel left aboard your aircraft when they reach the target. It is desirable to maximize the fuel left in order to engage or avoid enemy interceptor aircraft over the target or to attack secondary targets once the primary target, ONRODA airfield, has been destroyed. Your job is to help the computer come up with the best airstrike plan between the task force and the target within a specified time limit.

The best air strike plan minimizes the probability of the aircraft being detected by the radars and, at the same time, leaves maximum fuel remaining upon arrival at the target so that enemy fighter aircraft can be engaged or evaded.

The purpose of this material is to acquaint you with the:

1. Detection ability of multiple enemy radars when there is overlapping detection coverage between radars in proximity to each other

2. Means of measuring the goodness of an air strike plan

3. Characteristics of the dynamic programming optimization technique.

The training goals are to:

1. Develop expertise in using the equipment

2. Develop a feel for the best way to help the computerized technique find the best air strike paths and speeds.

In training you will do eight problems with the optimization technique. Experimental data collection will then be done for twelve problems. Thus, you will do a total of 20 problems. Each problem will last 15 minutes.

A. REPRESENTATION OF ENEMY RADAR DETECTION CAPABILITY

The capability of a single enemy radar to detect your aircraft is represented by concentric circles around the radar location. Detection capability is the same at all points on each circle and is a specified percentage of the peak detection capability of the radar. (See Figure D2.) Notice that as you go along a radial line toward the center of the concentric circles, detection capability increases up to the 90% of the peak level. The peak occurs between the two 90% circles and detection capability decreases from the peak as you get closer to the radar location. Thus detection capability may be visualized in three dimensions as a volcano with a rim and a crater in the center of the volcano. The "Detection volcano" is centered on the radar's position.

When several radars have overlapping coverage as shown in Figure D3, the probability of detecting your aircraft at a point within areas of overlap is higher than it would be at that same point if only one radar could detect at that point. Thus there is a joint detection capability throughout areas of overlap. The points where joint probabilities of detection are equal are connected together to form contours as shown in Figure D4. The contours have the same general meaning as the concentric circles in Figure D2, that is, each contour is the set of points where detection capability is some specified percentage of the peak joint detection capability. The set of contours is analogous to a topographical map. The difference is that each contour on a topographical map corresponds to an altitude above sea level and each detection capability contour corresponds to a detection capability between zero capability and the peak capability.

B. MEASURING THE GOODNESS OF AN AIR STRIKE PATH: THE UTILITY FUNCTION

The problem is to select an optimal air strike path, so an appropriate utility criterion function is one which measures the "goodness" of an air strike path. The two variables selected to determine the goodness of an air strike path were fuel consumption along the path and probability of being

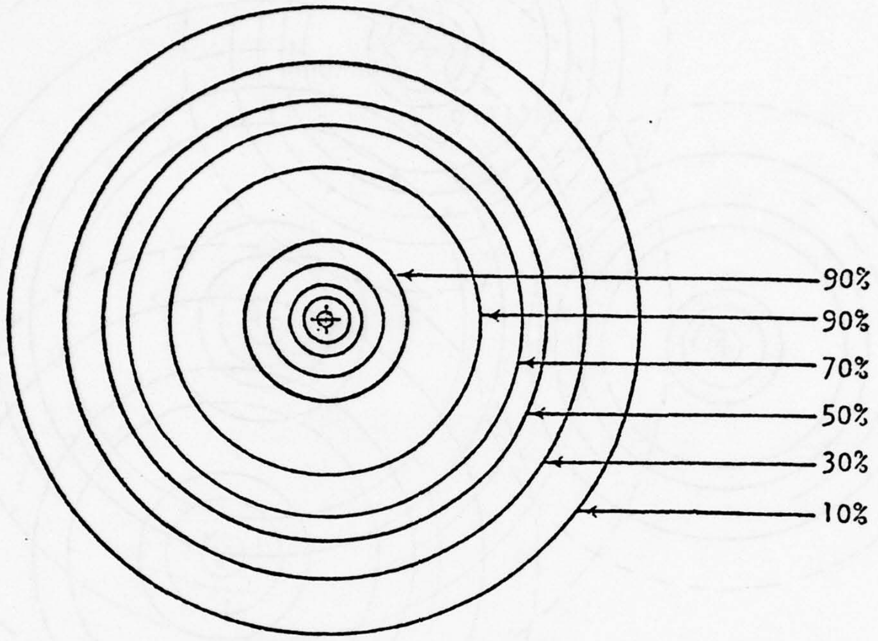


Figure D2. Single Sensor Coverage Template. (Circles show distance from sensor location, center, at which percentage values of the peak detection rate occur.)

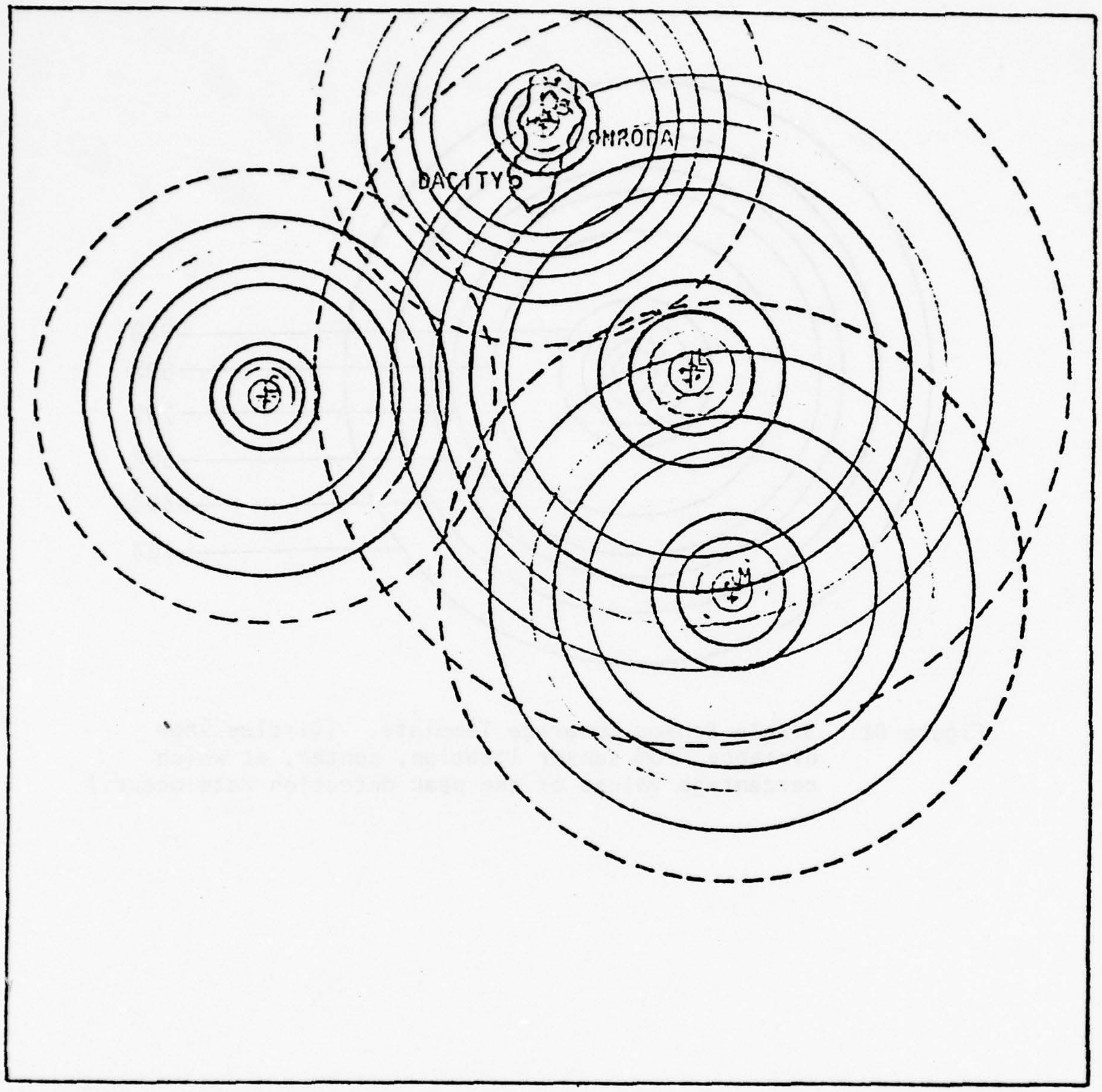


Figure D3. Display with all Sensor Coverage Templates Shown.

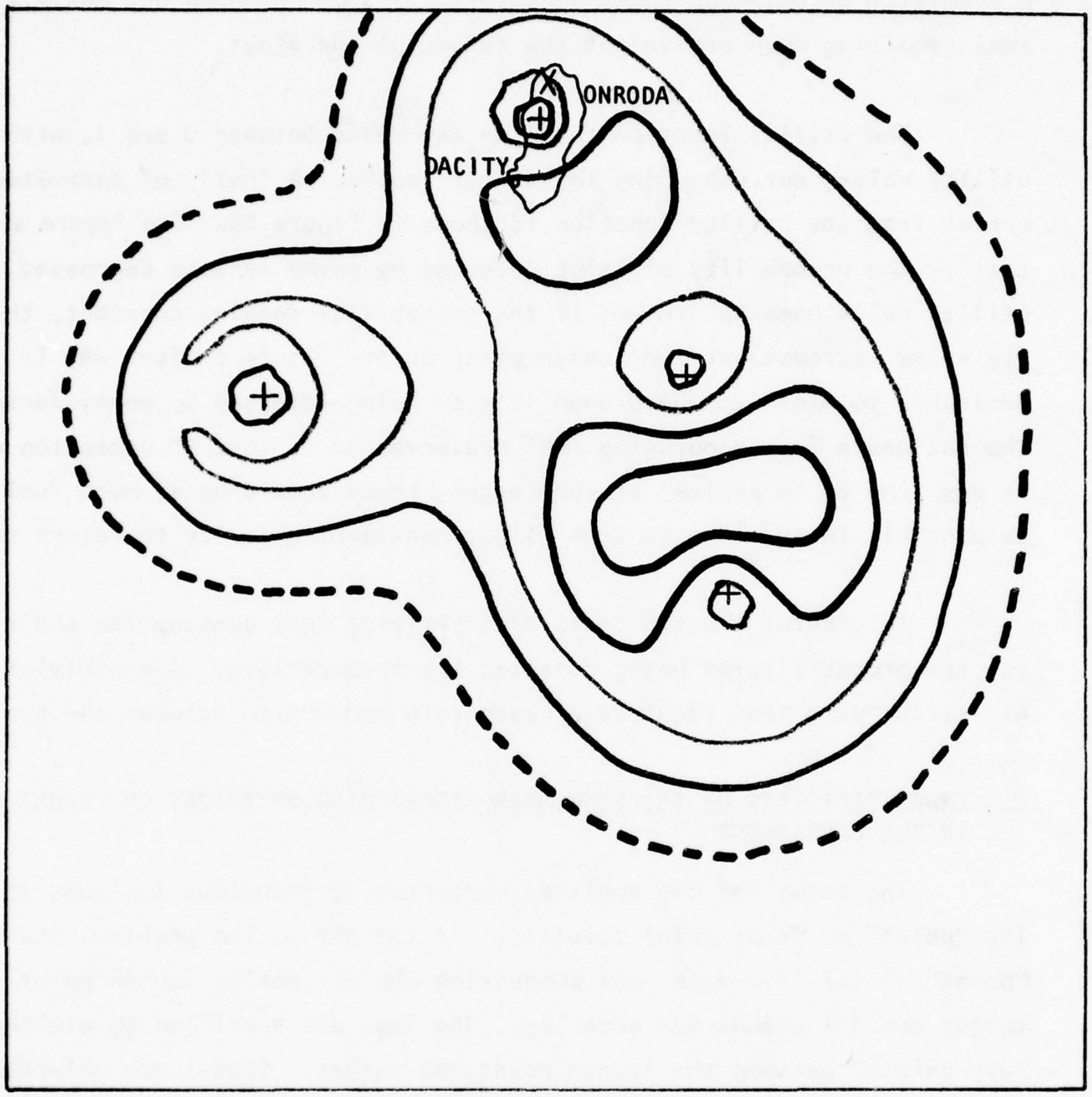


Figure D4. Contours Showing Joint Probabilities of Detection for Four Radars Located at Positions Marked with a Cross (+).

detected by one or more enemy radars prior to reaching the target. The utility criterion function incorporates a tradeoff between minimizing the probability of being detected by enemy radars on one hand and maximizing the fuel remaining upon arrival at the target on the other.

The utility function takes on any value between 0 and 1, with higher utility values corresponding to "better" paths. A family of parameterized curves from the utility function is shown in Figure D5. The figure shows that as the probability of being detected by enemy sensors decreases, the utility value goes up. Also, if the probability remains constant, the utility value increases as fuel consumption drops. It is obvious why it is desirable to minimize the probability of being detected by enemy sensors. The rationale for encouraging fuel preservation is that if detection occurs at any time up to arrival at the target, there should be as much fuel left as possible in order to do some flight maneuvering to try to return safely.

In general the two goals of minimizing fuel consumption and minimizing the probability of being detected are incompatible. A nontrivial optimal air strike path thus requires a reasonable compromise between the two goals.

C. CHARACTERISTICS OF THE NONLINEAR PROGRAMMING OPTIMIZATION TECHNIQUE USED IN THE EXPERIMENT

The setup for the nonlinear programming technique includes the starting "point" or first trial solution. In the air strike problem, the starting "point" is (a) five path legs connecting the air strike launch point and the target and (b) speeds for each leg. The legs are specified by picking four "way points" between the launch point and target. Speeds are selected from a range of 250 to 1000 knots. After the start point has been specified, the NP technique operates to find a better combination of way points and speeds. It does this by exploring changes in the location of each way point and the speed for each leg. Each exploration involves a single way point or a single speed. Therefore, improvement in the air strike path takes place slowly over many explorations, i.e., trials. An advantage of NP is that it considers all the points in a geographical region instead of just a set of grid points and all speeds instead of just a few. A disadvantage is that the "solution"

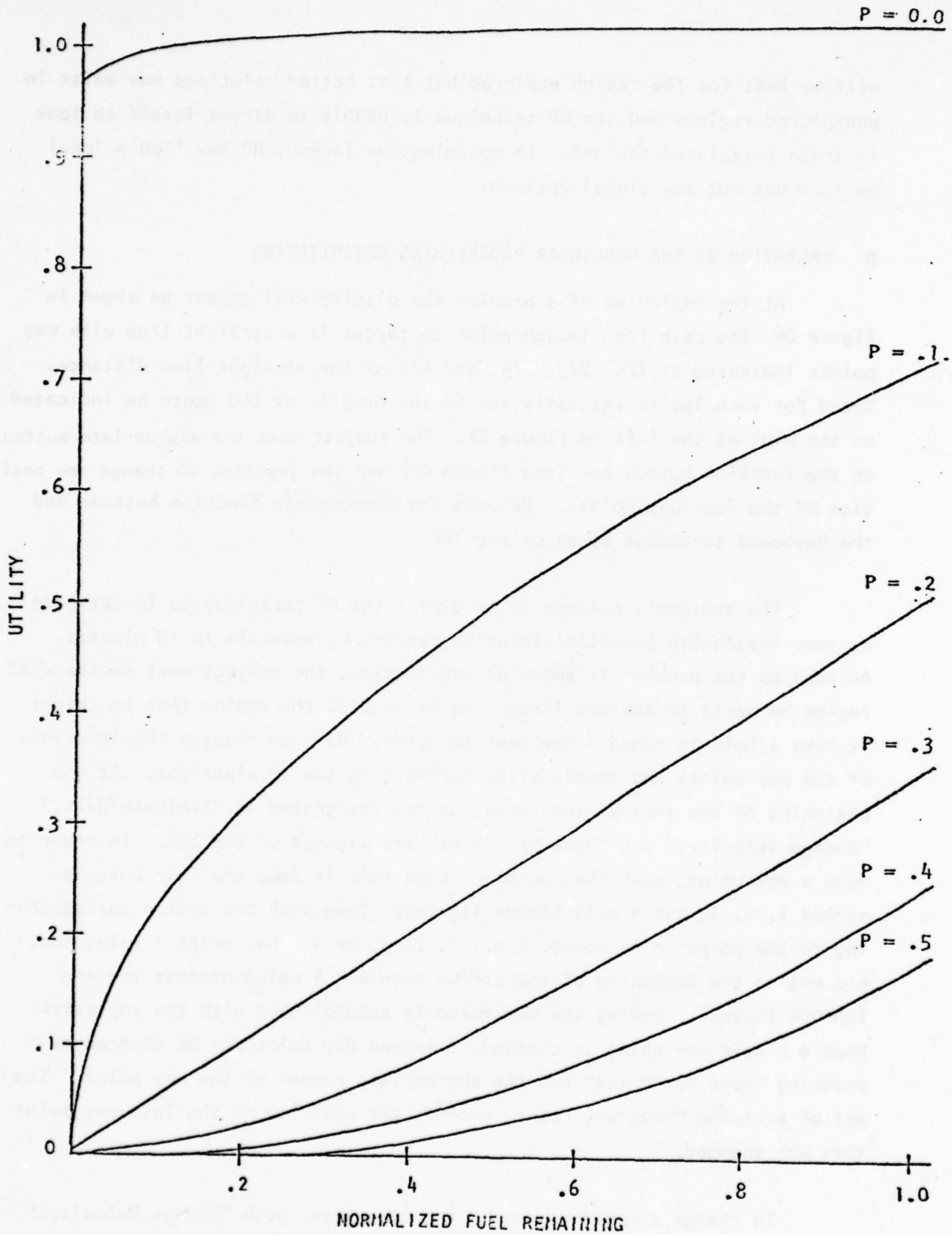


Figure D5. Family of Parameterized Utility Function Curves.

will be best for the region explored but that better solutions may exist in unexplored regions and the NP technique is unable to direct itself to look in these unexplored regions. In optimization jargon, NP may find a local optimum but not the global optimum.

D. OPERATION OF THE NONLINEAR PROGRAMMING OPTIMIZATION

At the beginning of a problem the display will appear as shown in Figure D6. The path from launch point to target is a straight line with way points indicated at 1/5, 2/5, 3/5, and 4/5 of the straight line distance. Speed for each leg is initially set by the program at 600 knots as indicated on the plot at the left in Figure D6. The subject uses the appropriate buttons on the function button box (see Figure D7) and the joystick to change the position of the four way points. He uses the appropriate function buttons and the keyboard to change speed on any leg.

The subject's purpose is to direct the NP technique to investigate as many reasonable potential solution regions as possible in 15 minutes. As soon as the problem is shown on the display, the subject must decide what region he wants to explore first. He is to pick the region that he thinks is most likely to contain the best solution. He then changes the locations of the way points and speeds prior to starting the NP algorithm. At the beginning of the problem the three buttons designated as "Evaluate/Halt," "Change Velocity," and "Move Way Point" are lighted on the box. In order to move a way point, push that button. When this is done the four buttons marked 1, 2, 3, and 4 will become lighted. Then push the button corresponding to the point to be moved, i.e., 1, 2, 3, or 4. Way point 1 being that closest to the beginning of the strike path and 4 being nearest the end (ONRODA Island). Moving the way point is accomplished with the joy stick. When a single way point is changed, a second way point can be changed by pressing "Move Way Point" and the appropriate number of the way point. The act of pressing "Move Way Point" records the position of the last way point that was changed.

To change a speed on one of the five legs, push "Change Velocity." The five buttons marked 1, 2, 3, 4, and 5 will light. Leg 1 refers to the

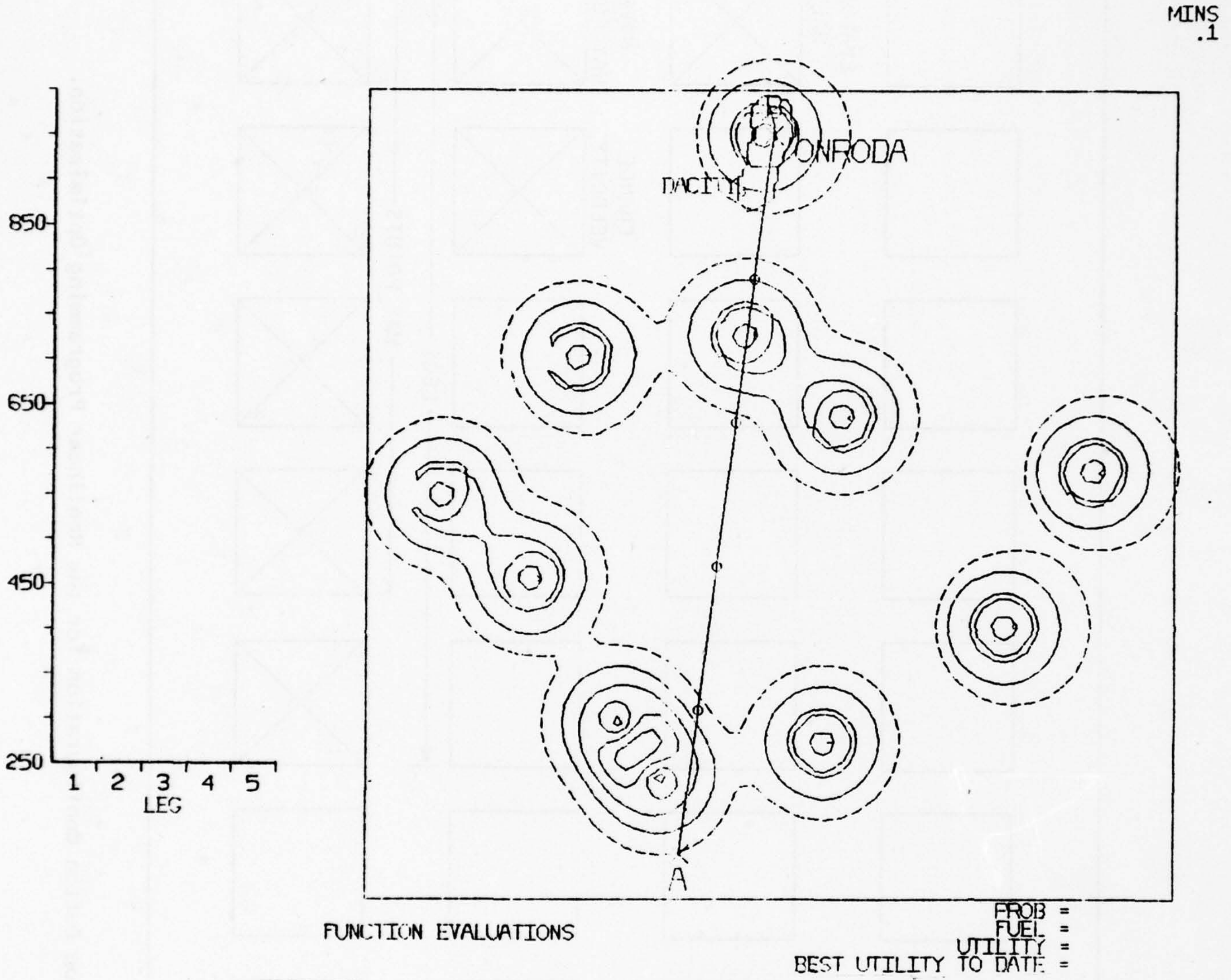


Figure D6. Display Appearance at Beginning of Problem to be Solved Using NP Algorithm.

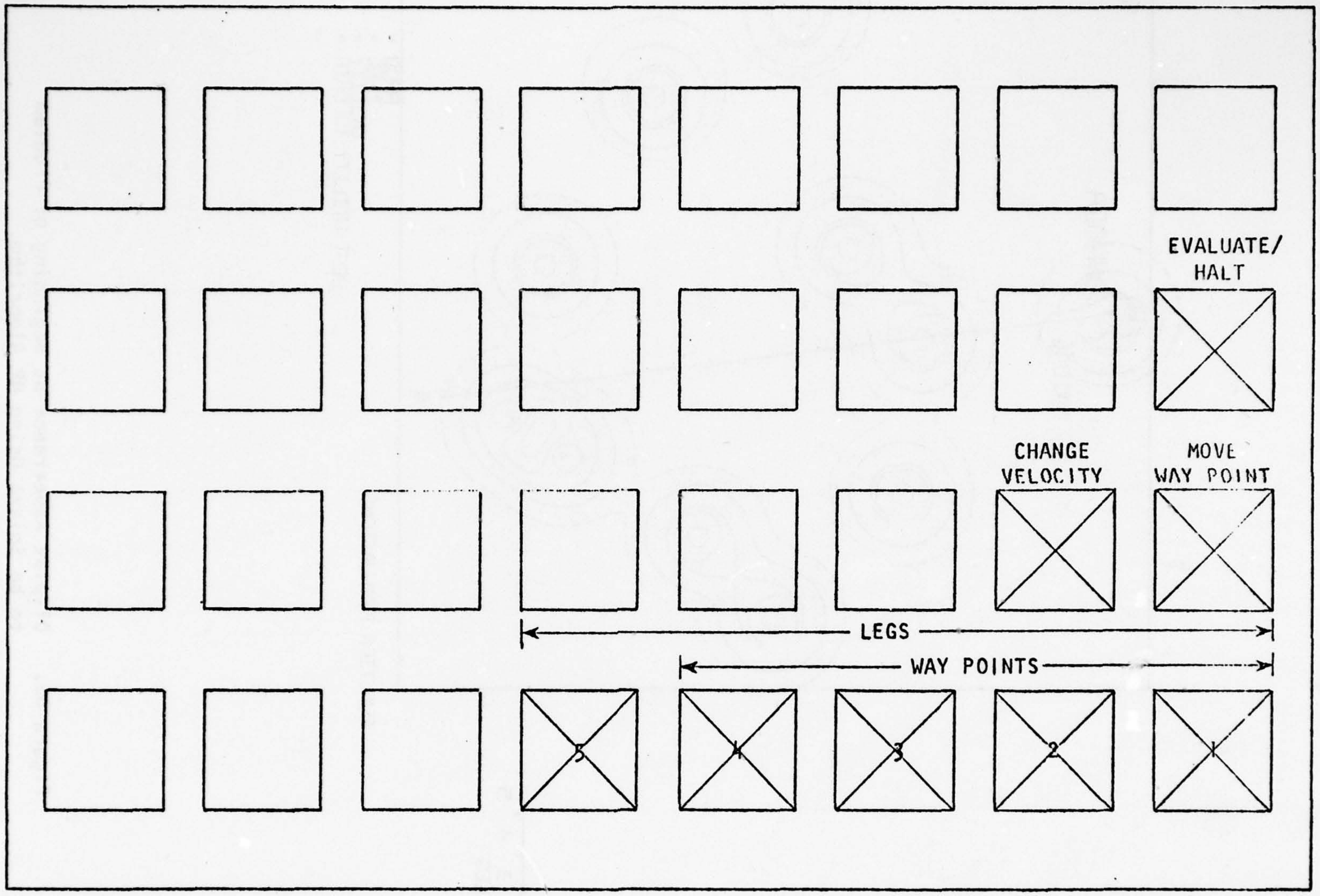


Figure D7. Function Button Configuration for the Nonlinear Programming Optimization.

leg closest to the path beginning point and leg 5 refers to the path closest to the end point (ONRODA Island). Then push the button corresponding to the leg for which you want to change speed and:

1. Use the teletype keyboard to input the speed you want used on the selected leg. Put a decimal point at the end of the number. (This is essential.)
2. Push the teletype key marked "CR."

Thus, if you wanted to change the speed on leg 3 to 850 knots, you would:

1. Press function button "Change Velocity"
2. Press function button "3"
3. Press teletype key "2"
4. Press teletype key "0"
5. Press teletype key "0"
6. Press teletype key "."
7. Press teletype key "CR"

When you have changed all the way points and speeds to those you want, then press the function button marked "Evaluate/Halt." The NP algorithm will begin to operate, i.e., "Evaluate," using your starting point consisting of the four way points and five speeds. Once the algorithm has begun operating, only the "Evaluate/Halt" button will remain lighted and the only control at the operator's disposal is to halt operation by pushing this button.

The primary indicators that the operator uses to decide to halt the algorithm are the displays of the number of function evaluations and the utility of the latest trial solution. In general, a plot of utility versus function evaluations would appear as shown in Figure D8. The subject should stop the algorithm when it reaches the point shown in Figure D8 because there will be little more utility to be gained by letting the algorithm continue. He should then input a new set of way points and speeds and start the algorithm again.

As the algorithm operates you will note variable length arrows appearing briefly at each way point. These represent potential changes in the location of a way point being considered by the algorithm. When utility

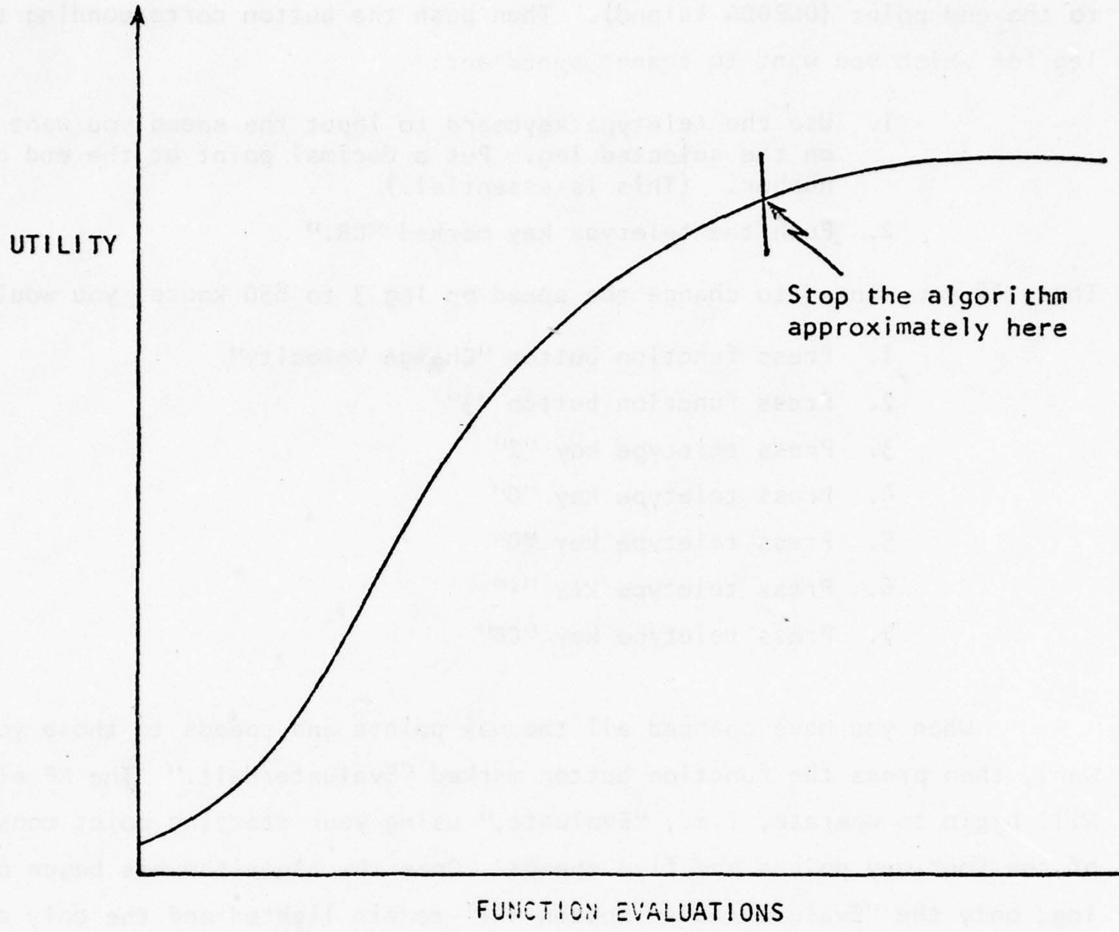


Figure D8. Typical Plot of Utility Versus Function Evaluations.

levels off, the magnitudes of changes in the following will also become small:

1. Value of "Prob," i.e., the probability that the air strike will be detected prior to arrival at the target.

2. Value of "Fuel," i.e., the fuel that will be consumed for the latest trial solution.

3. Speed changes indicated on the speed/leg graph.

4. Lengths of arrows appearing at each way point.

While the algorithm is operating on the first set of way points and speeds input by the operator, he should count the number of regions that could reasonably be expected to contain the best path. Dividing 15 minutes by the number of regions to be explored indicates approximately the number of minutes the operator should devote to each region. Depending on the problem, there will be enough time to explore 3, 4, or 5 regions.

At the end of 15 minutes the computer will have stored:

1. The utility of the path comprised of the first way points and leg speeds.

2. The utility of each best-solution-to-date at the end of each minute.

These are the data that will be used in the analysis of operator generated data.

E. GUIDELINES

There are two types of data being analyzed:

1. Utility of the path comprised of the first way points and leg speeds. Thus the operator's first goal is to do the best he can on this.

2. Operator performance will be calculated at the end of each trial by adding the 15 utilities of the best-solution-to-date at the end of each minute and dividing this sum by 15. Thus, operator performance

for the entire trial is the average of the 15 utilities. The operator's second goal is to maximize this average. In general this is done by exploring the regions which could contain the best path in the order of estimated likelihood that each contains the best path. This is compatible with the operator's first goal because, if the operator is correct concerning the region which contains the best path, then the average utility will be nearly equal to the utility of the best path. This is true because the computer only stores the best utility to date and will therefore not store the utilities of paths investigated after the first when the first region explored contains the best path.

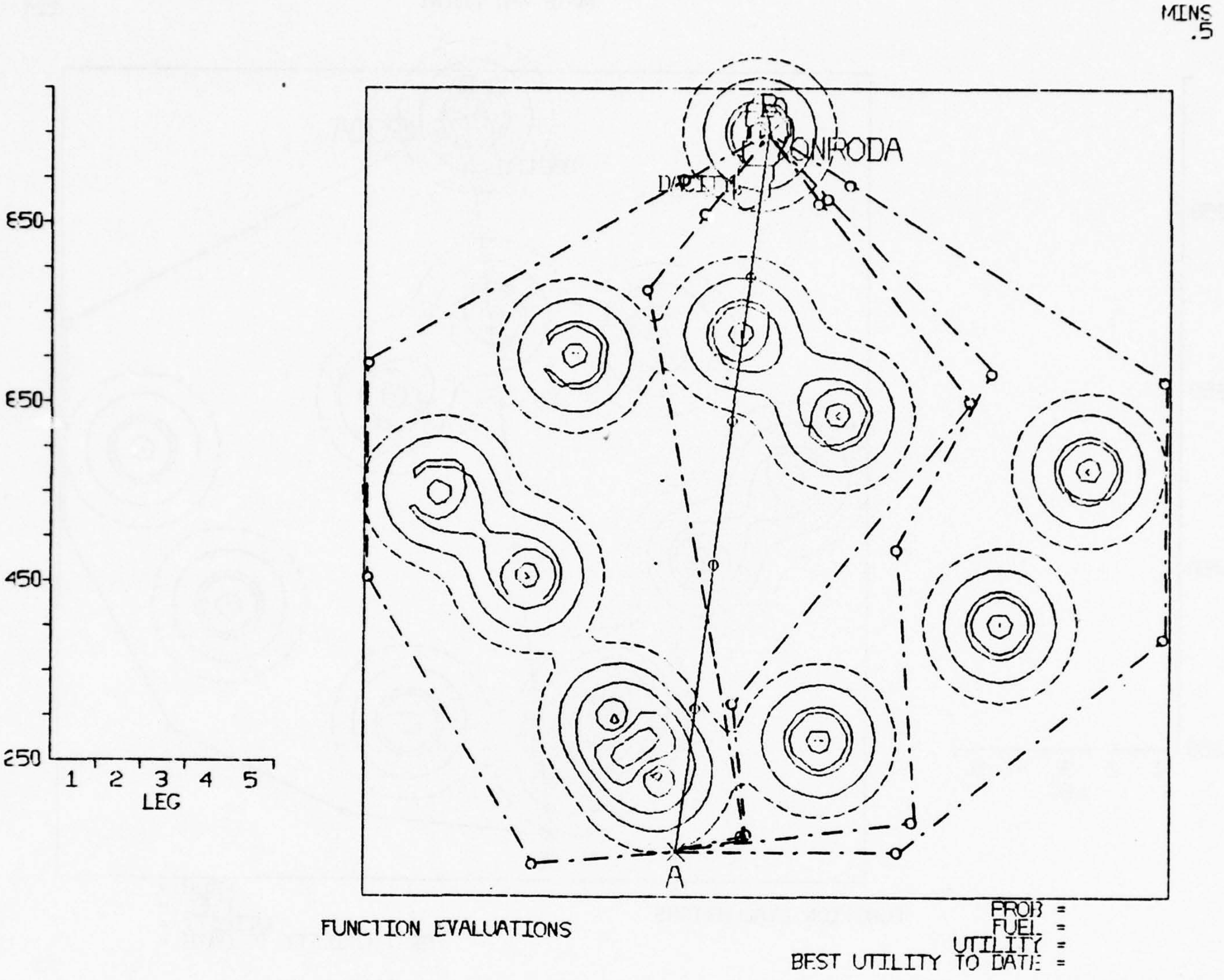
Other general rules to be used with the NP technique are:

1. Those portions of a path that are completely outside the detection contour should be transited at low speeds.

2. Those portions of a path that traverse a high detection probability contour should be transited at high speeds. In particular, the last leg of the path to the target should be transited at high speed since it must pass through the high detection region around ONRODA airport.

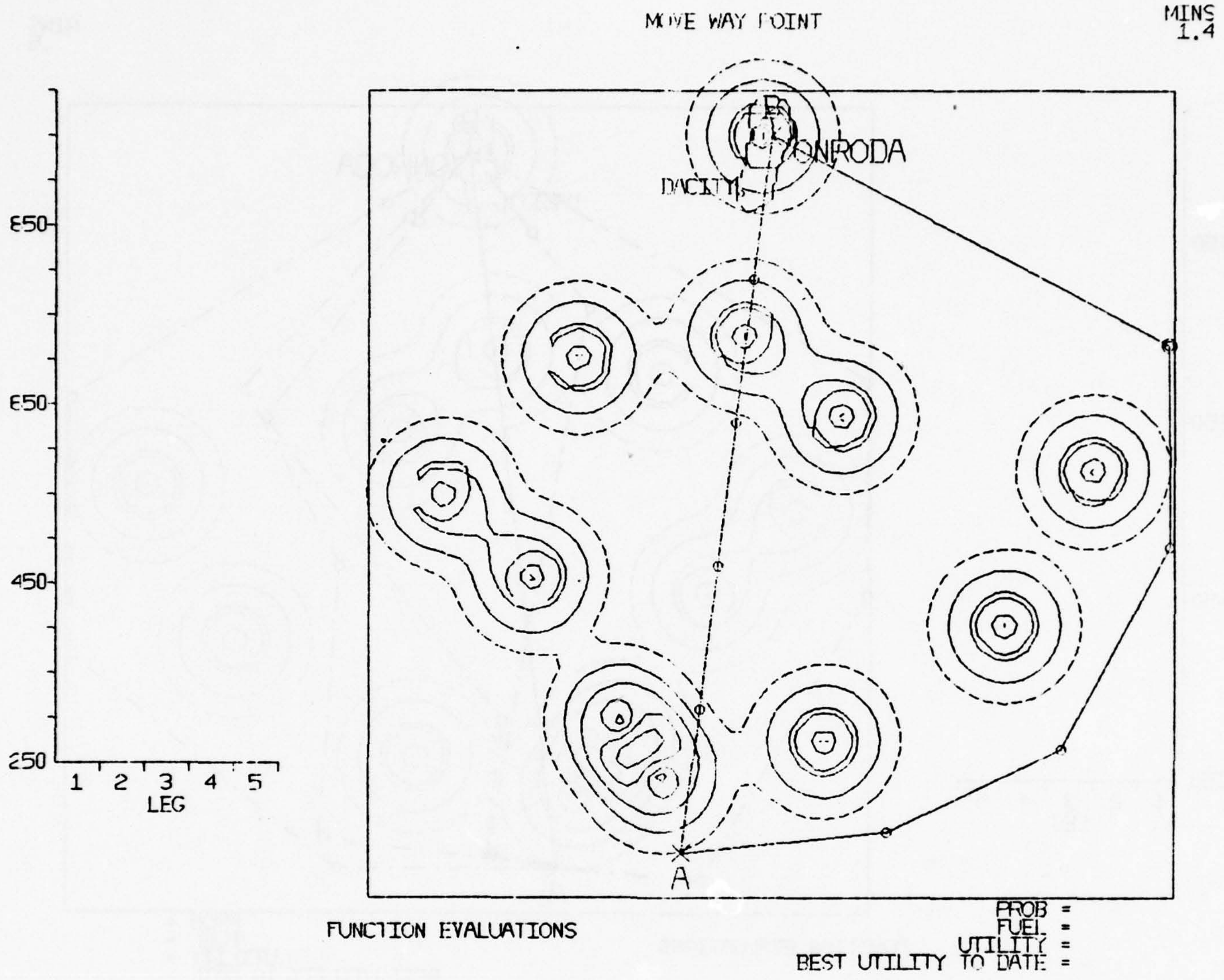
3. Paths should be drawn to pass through low detection probability regions. However, a completely roundabout path that avoids detection contours completely is not a sure winner because long paths use a lot of fuel.

4. When locating NP way points and the path will cross from outside the lowest detection probability contour to inside it, place one way point just outside the contour. If the path is crossing from inside the lowest detection contour to outside it, place a way point just inside the contour. These tactics will save computer time.



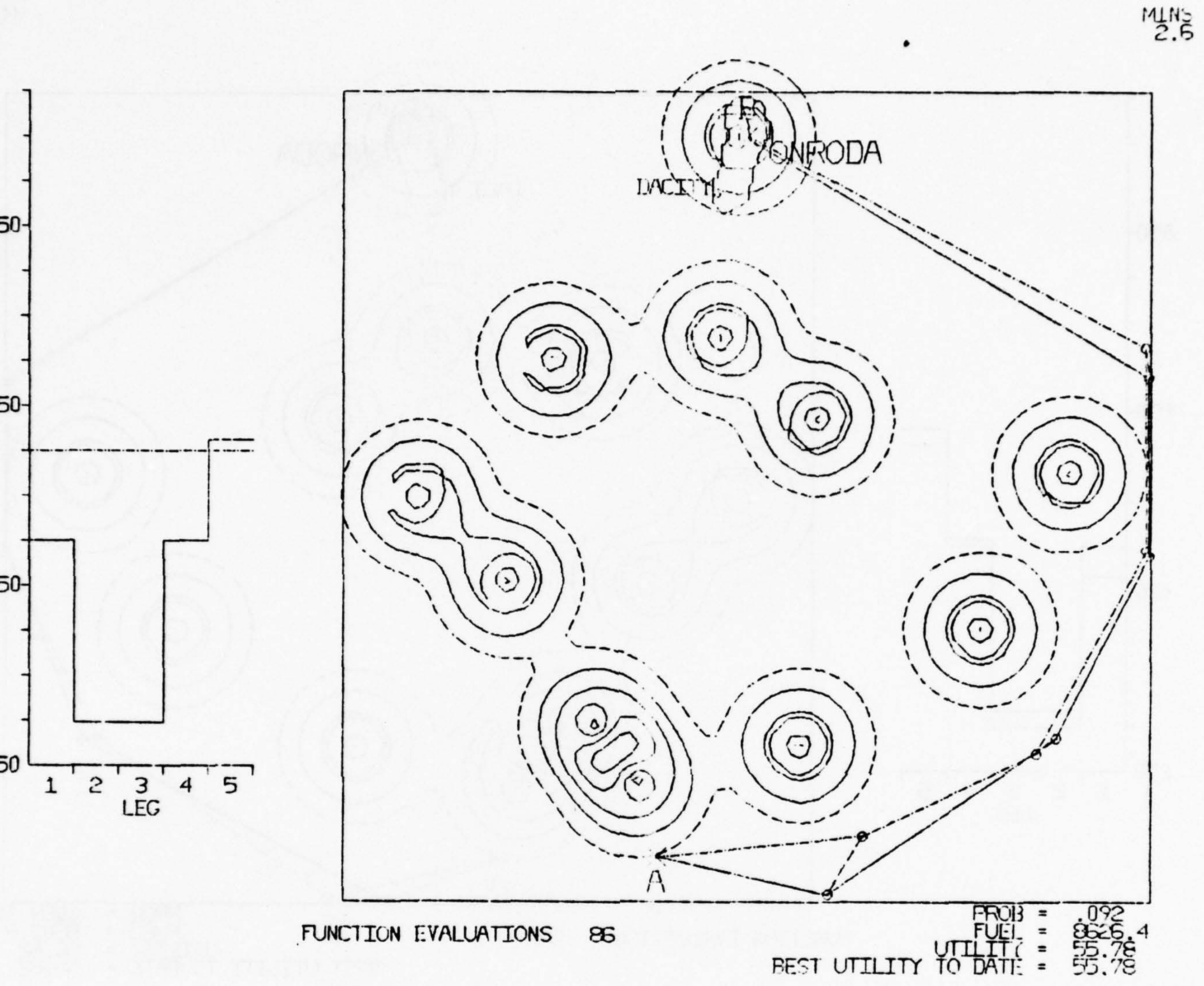
This is how the display appears at the beginning of the problem. Five potential best paths are shown as dot-dash-dot lines.

Figure D9. First Plate, Example Problem.



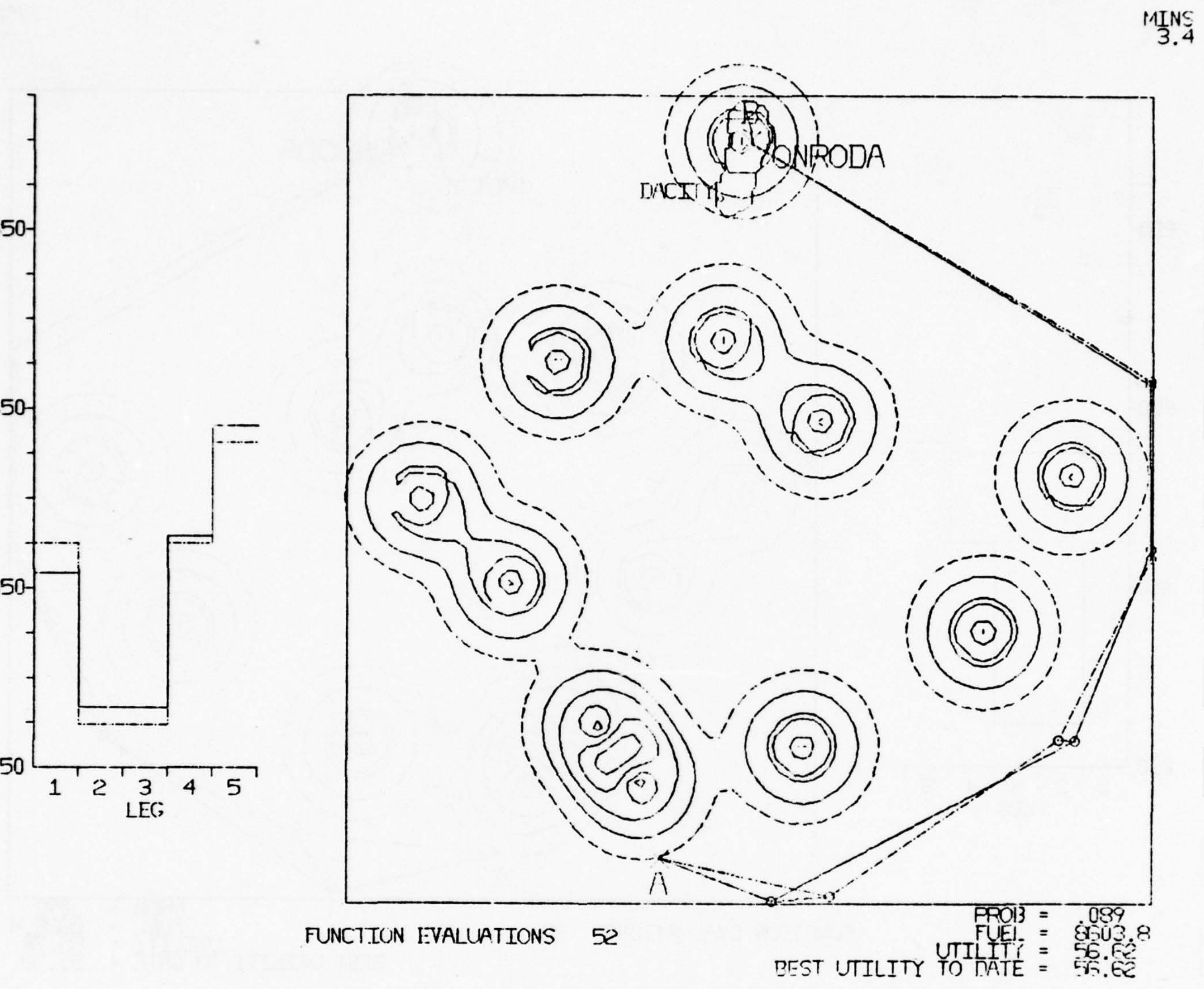
The operator chose to explore paths from right to left. It would have been better to have configured the path so that the last leg began just outside the contours around ONRODA. The previous starting path remains on the display as a dot-dash-dot line.

Figure D10. Second Plate, Example Problem.



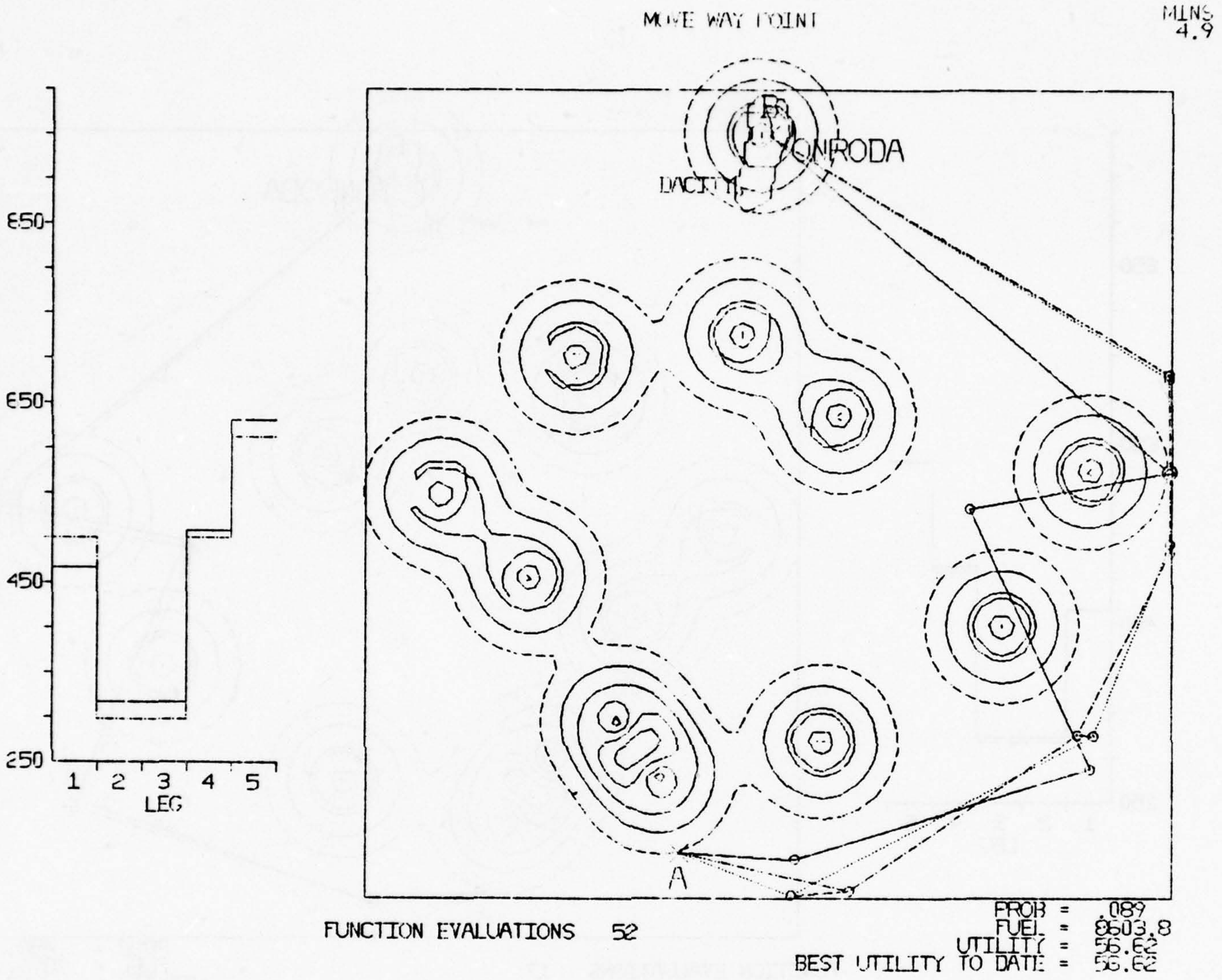
The operator stopped the algorithm at the end of 86 evaluations in order to get this picture. Note that the solution moved the first way point <u>down</u> in order to get away from the contours above the point.

Figure D11. Third Plate, Example Problem.



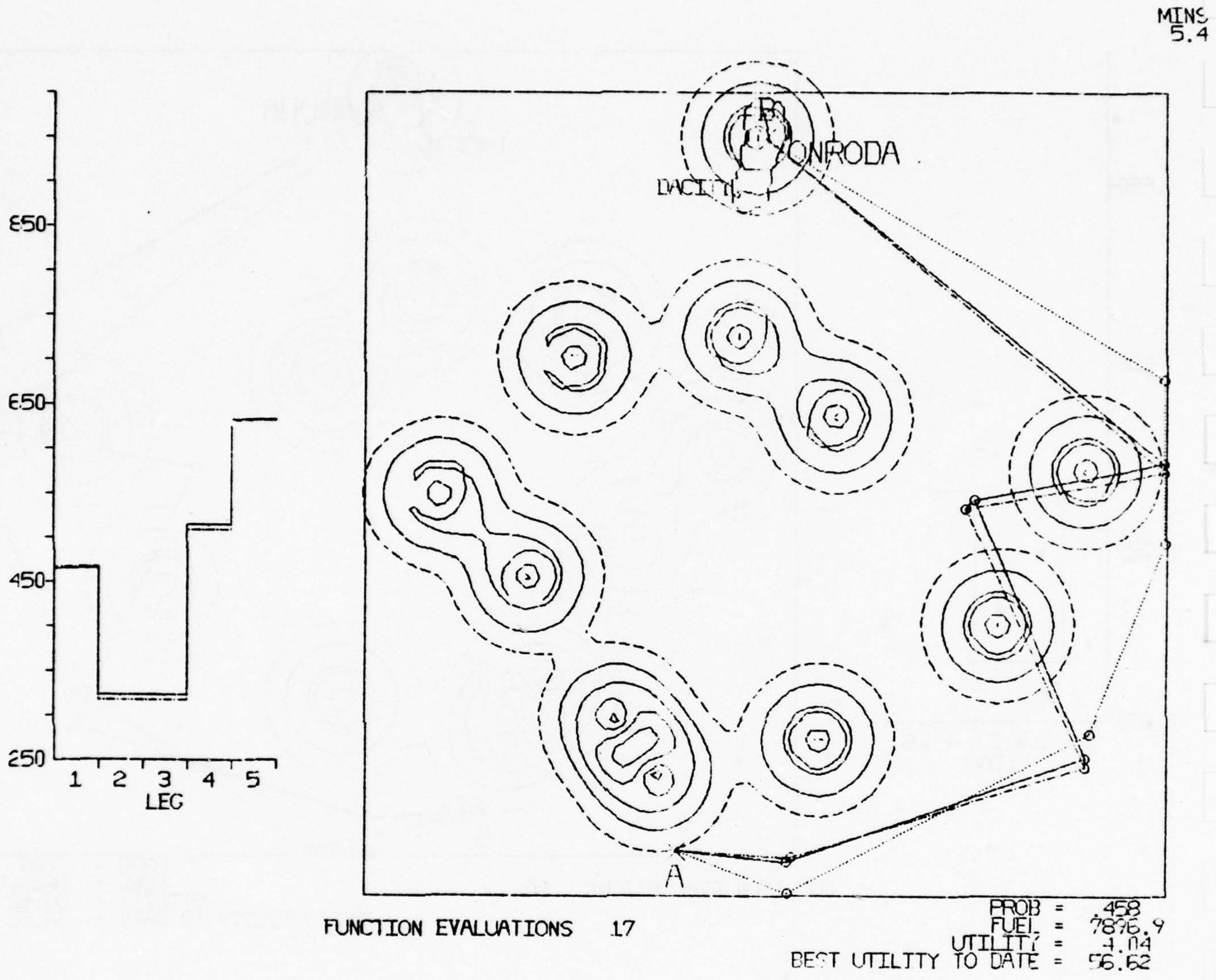
The operator restarts the algorithm without making any changes. At the end of another 52 evaluations (138 total), the operator stops the algorithm because (a) the step sizes being considered are very small and therefore the possible utility improvements will also be small, and (b) the utility hasn't increased very much in the last 25 or so evaluations.

Figure D12. Fourth Plate, Example Problem.



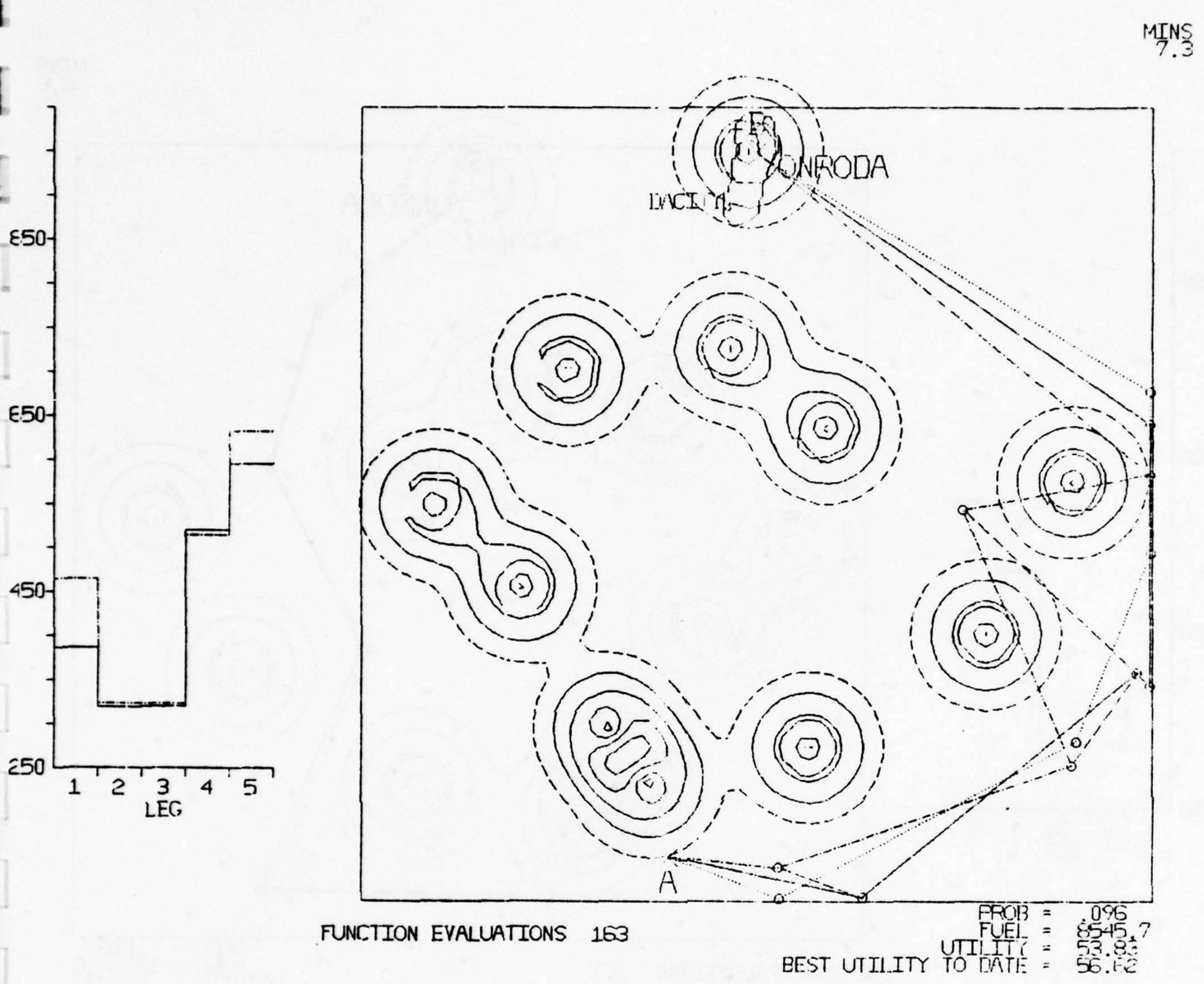
The operator puts the third and fourth way points in an illogical combination of places and makes small adjustments to the other two way points. The point will be to see what the algorithm does.

Figure D13. Fifth Plate, Example Problem.



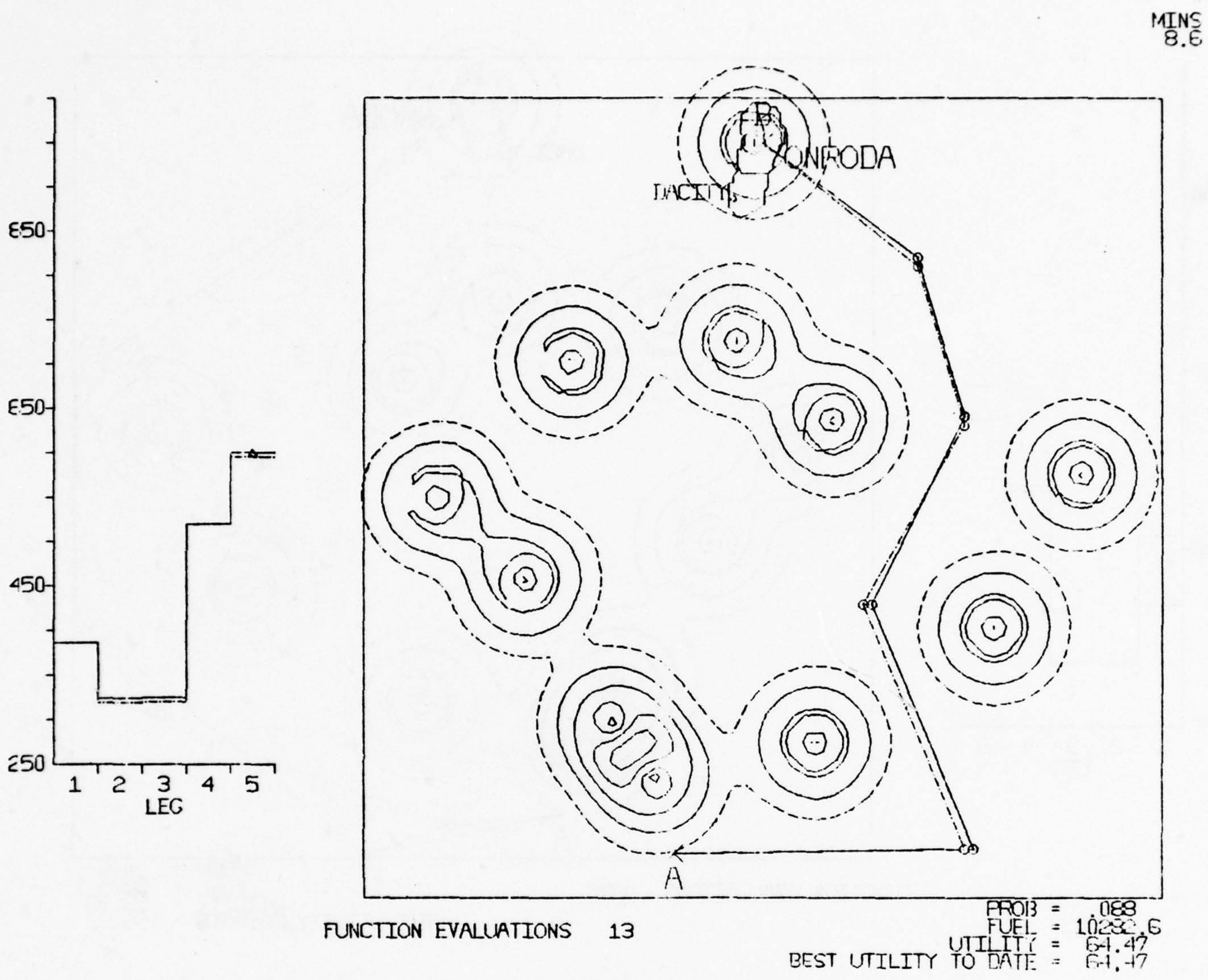
At the end of only 17 evaluations not much has happened.

Figure D14. Sixth Plate, Example Problem.



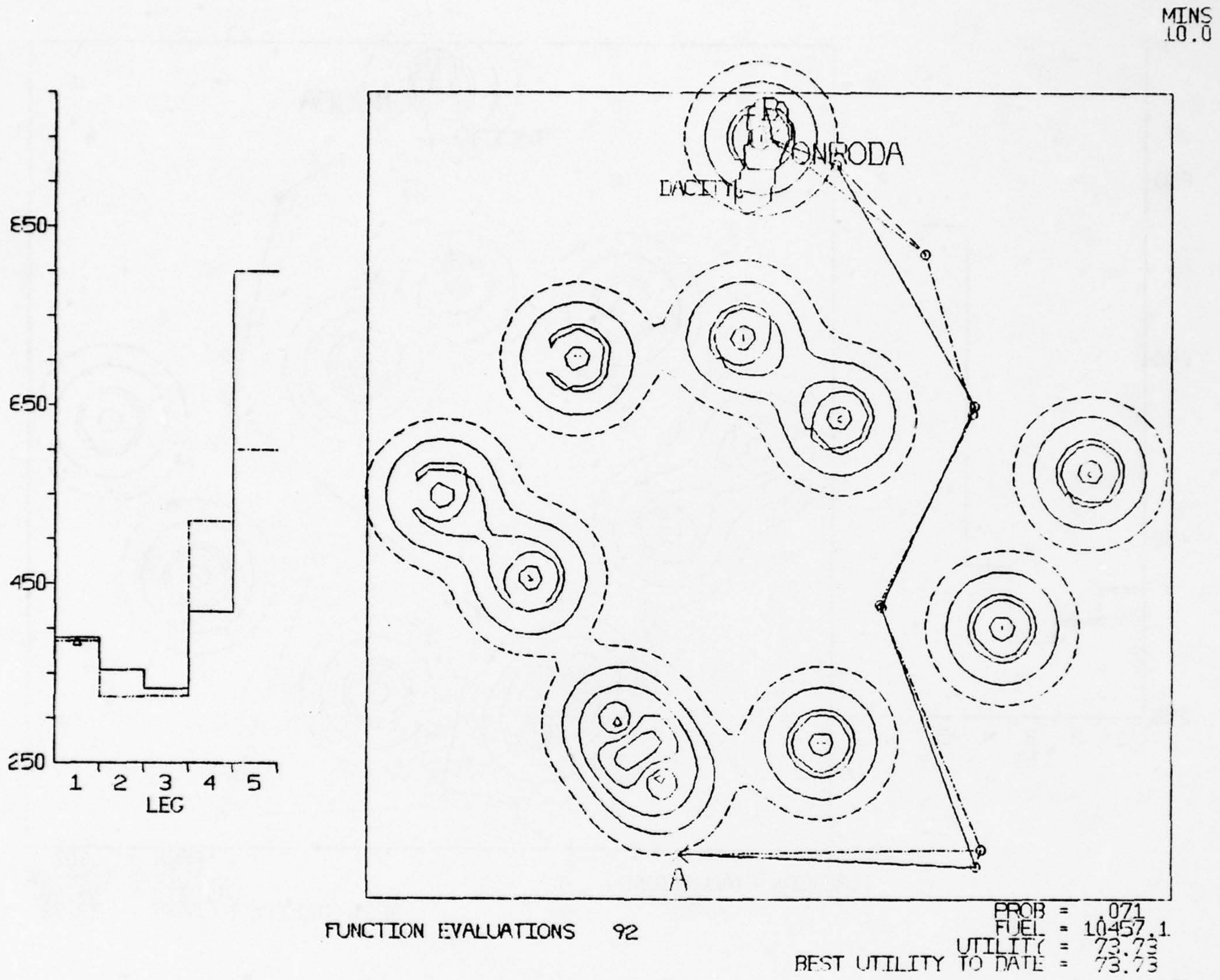
At the end of 163 evaluations the algorithm has found its way over to a much better position for the third way point but the utility is not as good at 163 evaluations (53.83) as it was at 140 evaluations with the earlier, better selection of way points (56.62 for the starting path of Figure D10).

Figure D15. Seventh Plate, Example Problem.



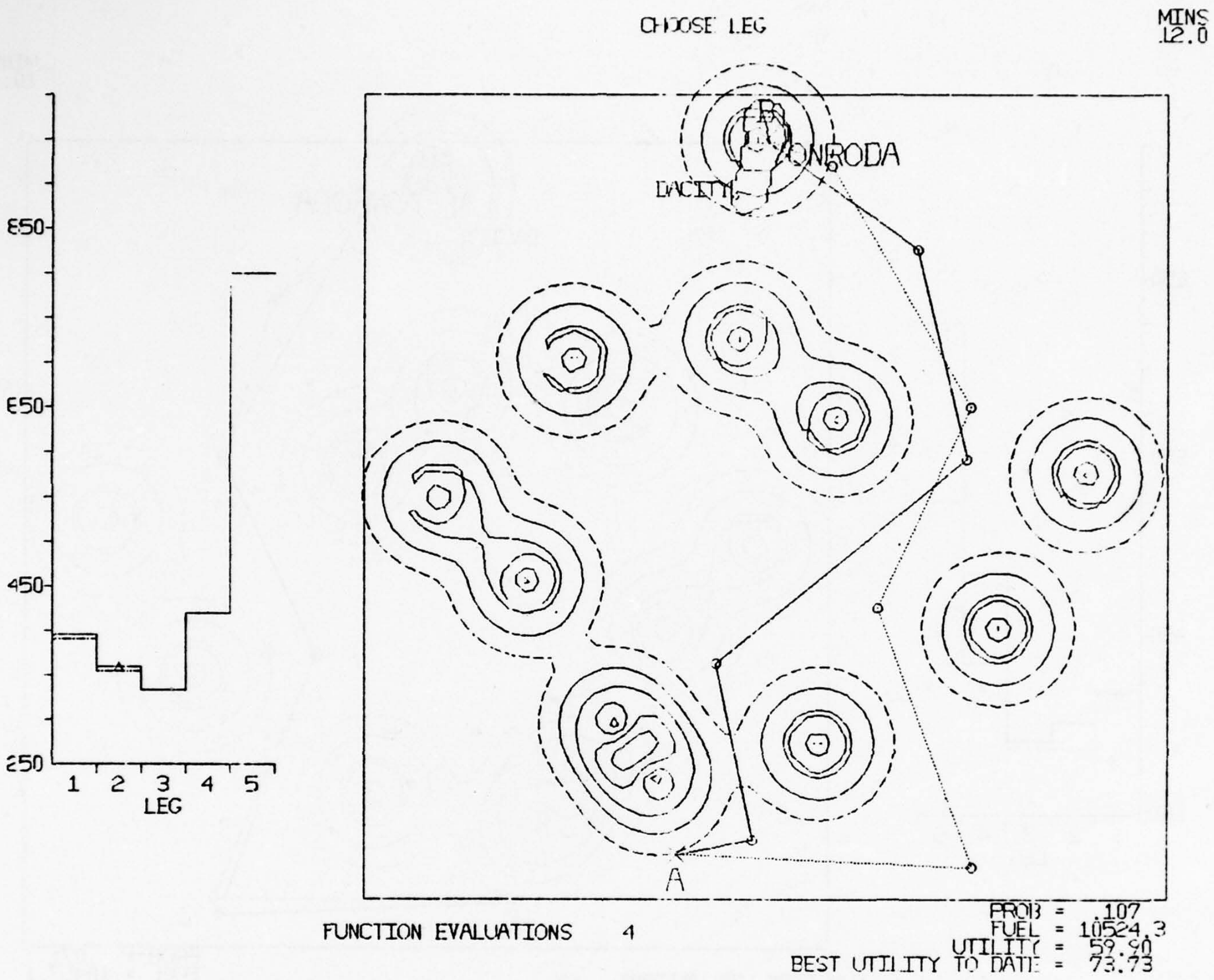
The operator selects a new set of way points and the algorithm begins to explore around these. Again, he should have placed the last way point closer to ONRODA.

Figure D16. Eighth Plate, Example Problem.



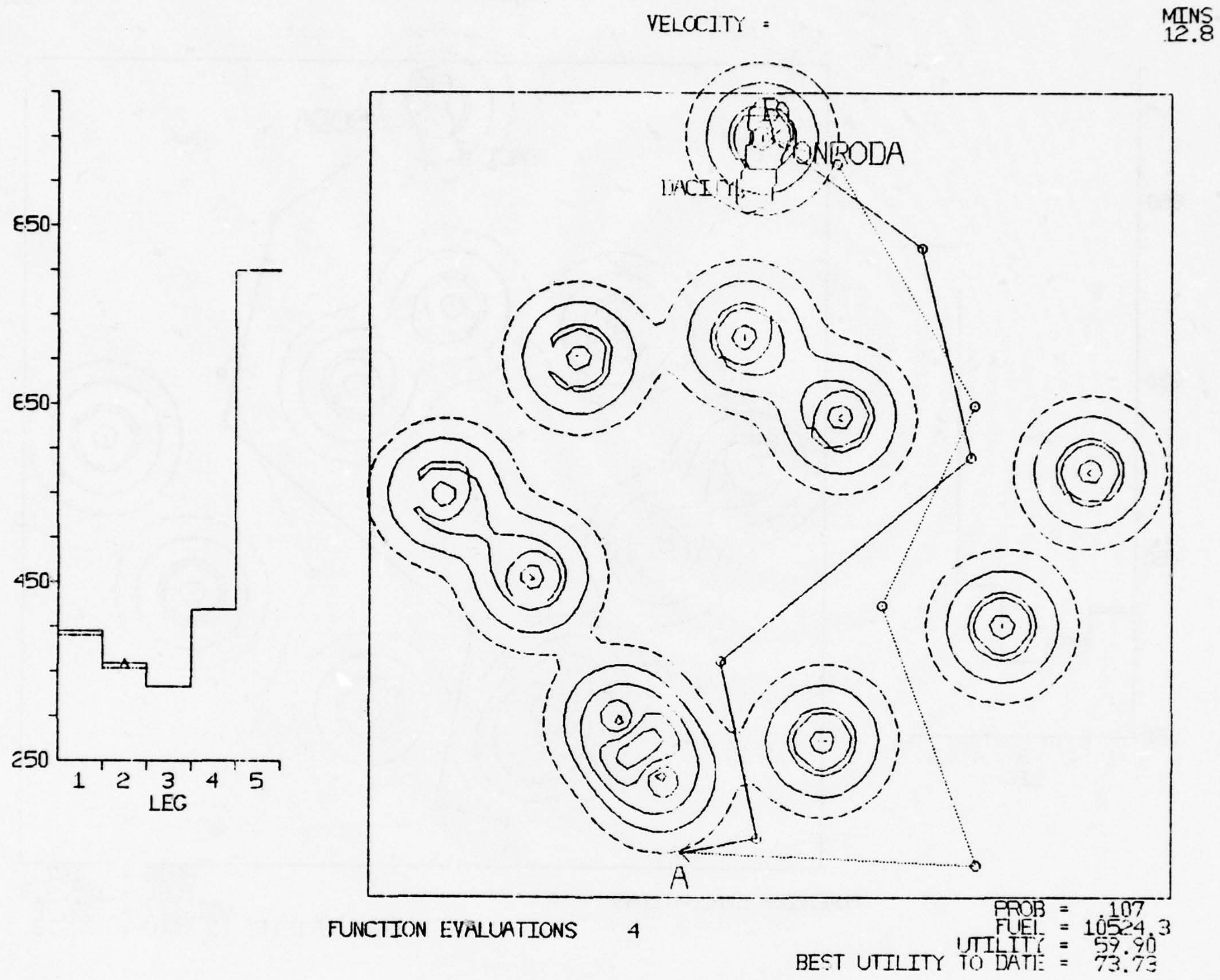
At the end of 92 evaluations the operator stops the algorithm. Note that the algorithm has moved the last way point much closer to ONRODA and has greatly increased the speed for the last leg.

Figure D17. Ninth Plate, Example Problem.



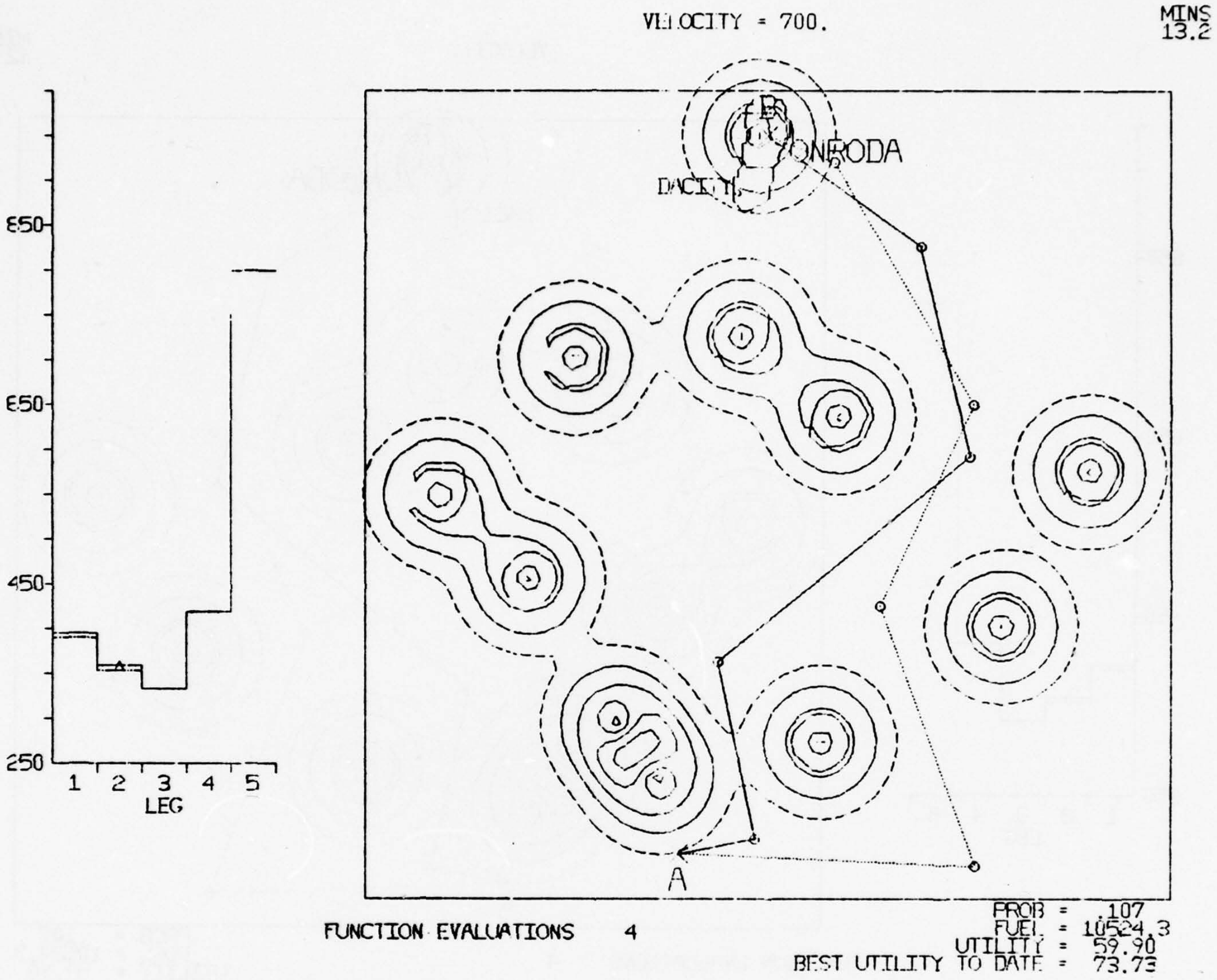
The operator has already selected way points for the third path to be explored by the algorithm. Utility is 59.90 at the end of four evaluations, and then the operator stops the algorithm. He has decided to change the speed on a particular leg and accordingly pushed the "Change Speed" function button. The prompt "Choose Leg" then appears at the top of the display. Then he pushes the function button corresponding to the desired leg.

Figure D18. Tenth Plate, Example Problem.



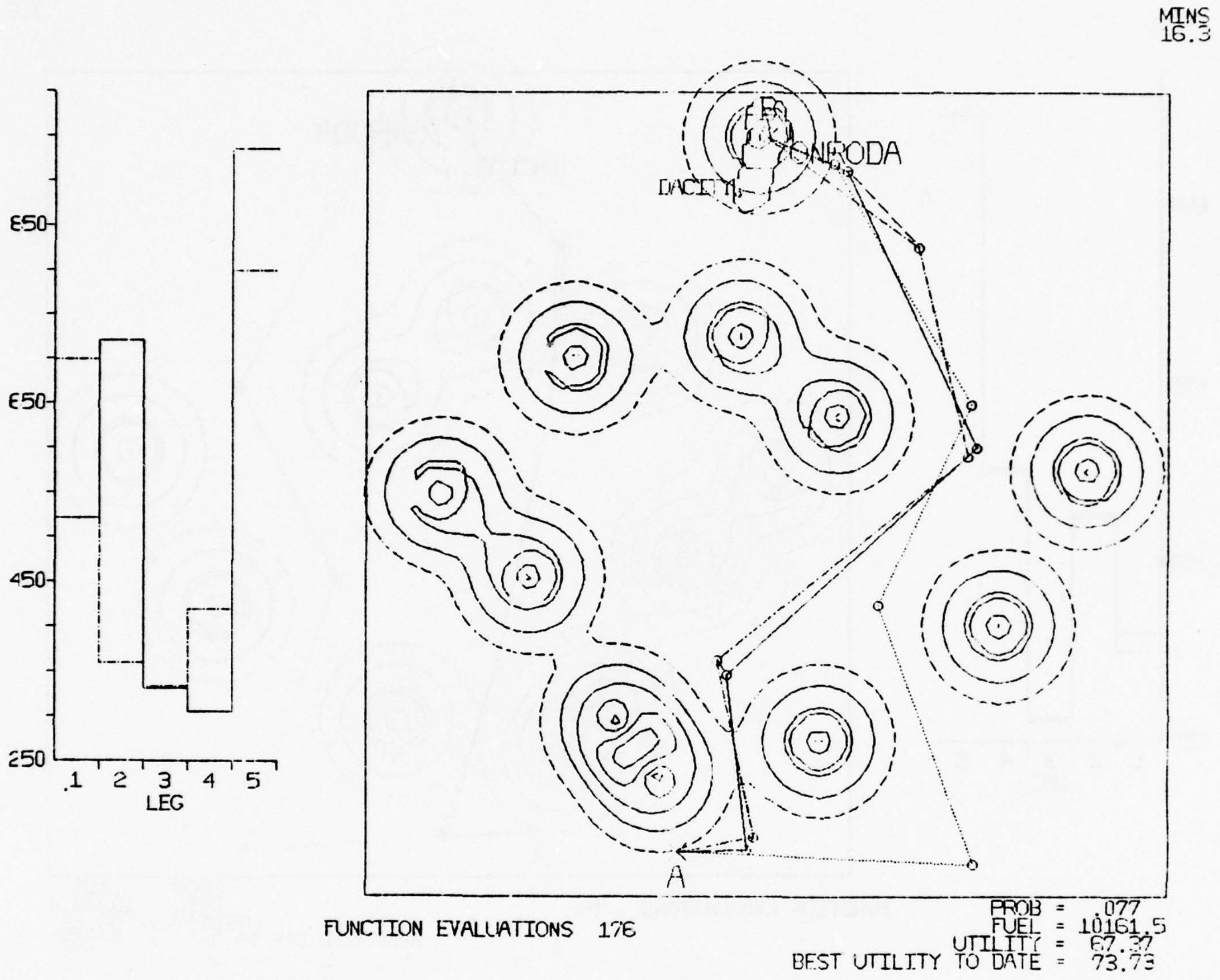
Immediately the prompt "Velocity = " appears at the top of the display.

Figure D19. Eleventh Plate, Example Problem.



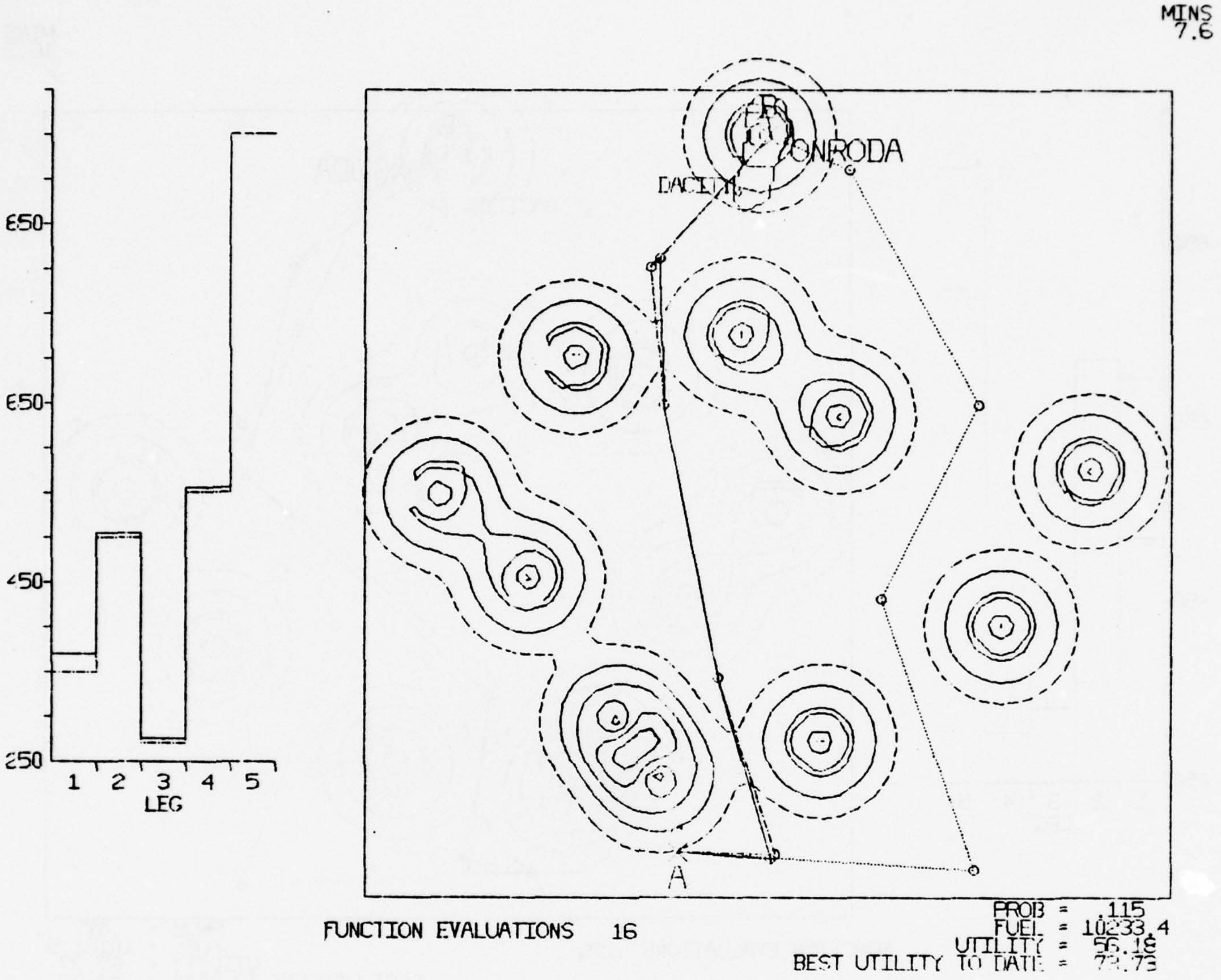
The operator then types "700. CR " and 700 appears at the top of the display. The operator restarts the algorithm.

Figure D20. Twelfth Plate, Example Problem.



The operator stops the algorithm at the end of 176 evaluations. Note again that the algorithm has moved the last way point much closer to ONRODA. (Disregard time shown under "MINS" from this figure on.)

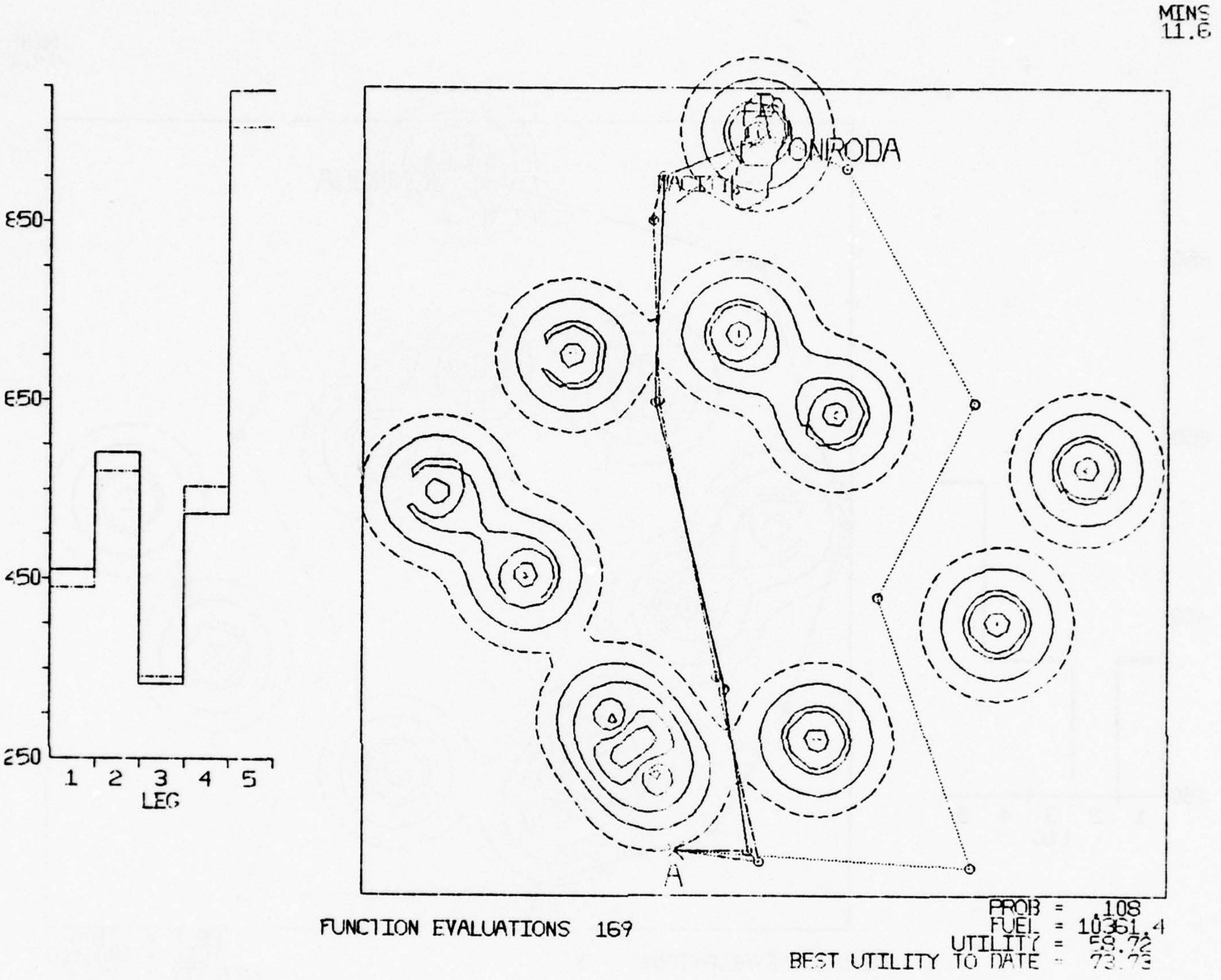
Figure D21. Thirteenth Plate, Example Problem.



```
FUNCTION EVALUATIONS   16              PROB = .115
                                       FUEL = 10233.4
                                    UTILITY = 56.18
                      BEST UTILITY TO DATE = 73.73
```

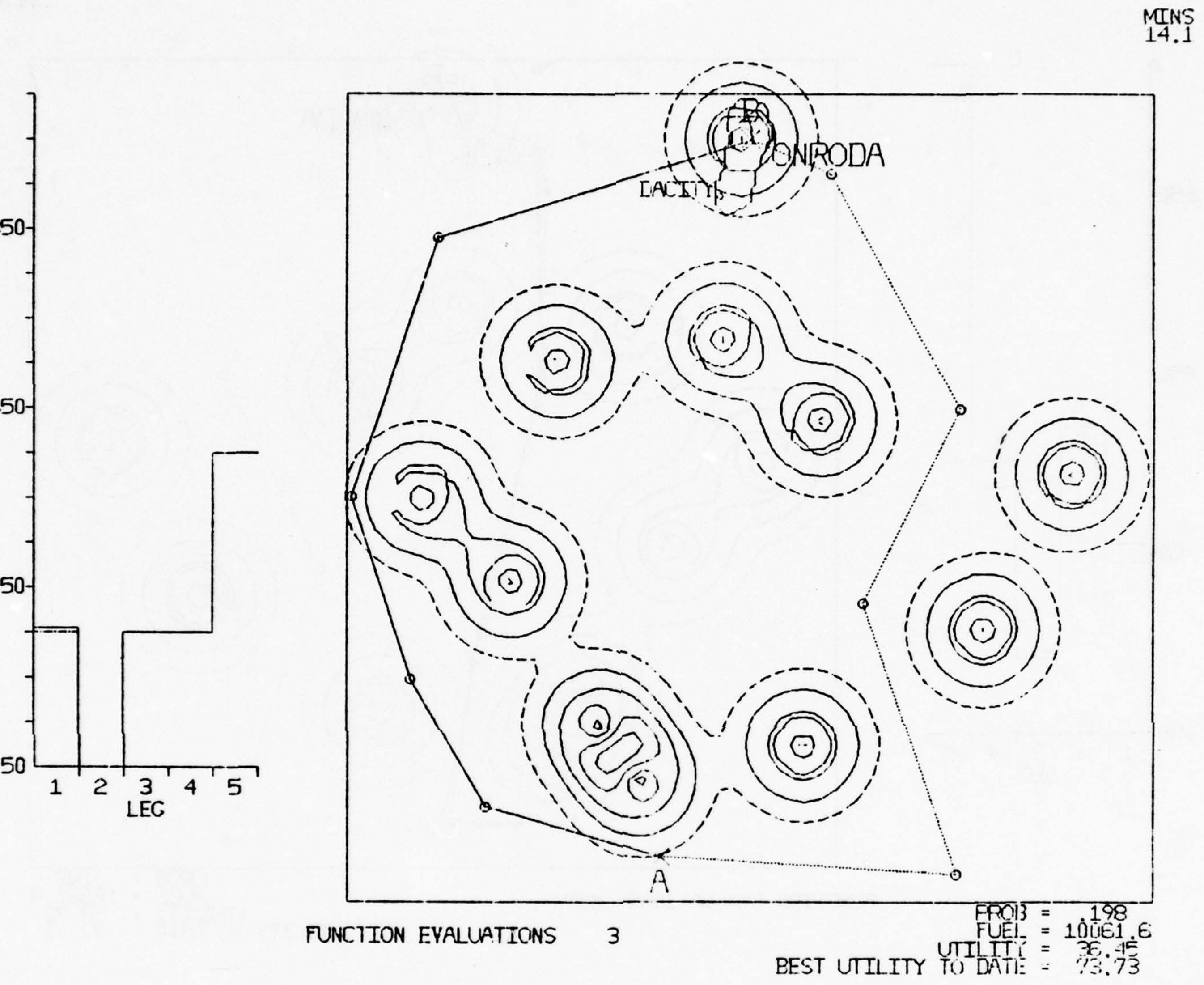
The operator has selected way points for exploring the fourth path and started the algorithm. At the end of 16 evaluations the utility is 56.18.

Figure D22. Fourteenth Plate, Example Problem.



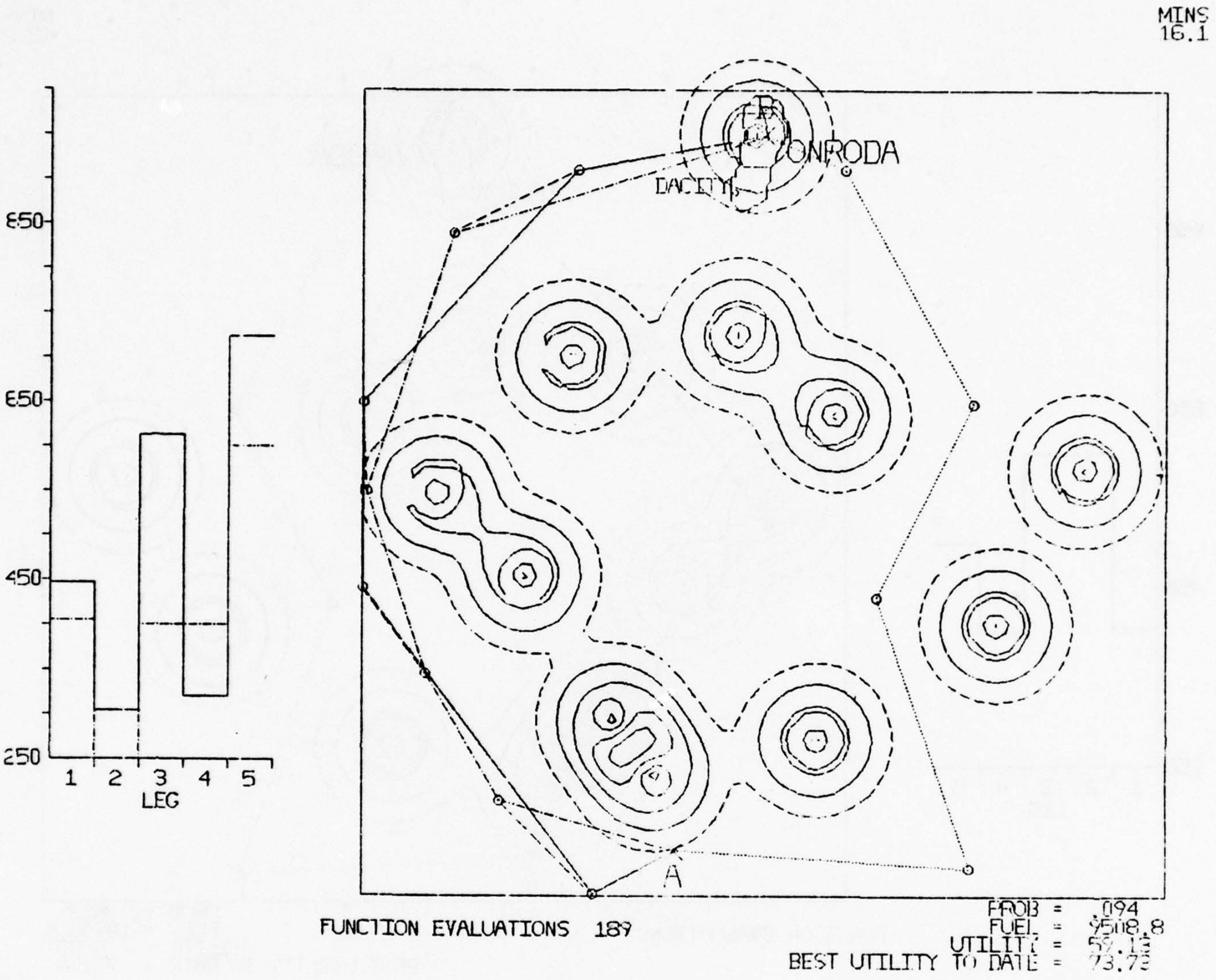
The operator stops the algorithm after 169 evaluations. Again, note that the algorithm moved the last way point closer to ONRODA. Utility is competitive with the utility for the first path explored (58.72 versus 56.62) but is significantly lower than the utilities achieved for the second and third paths explored (58.72 versus 73.73 and 67.37).

Figure D23. Fifteenth Plate, Example Problem.



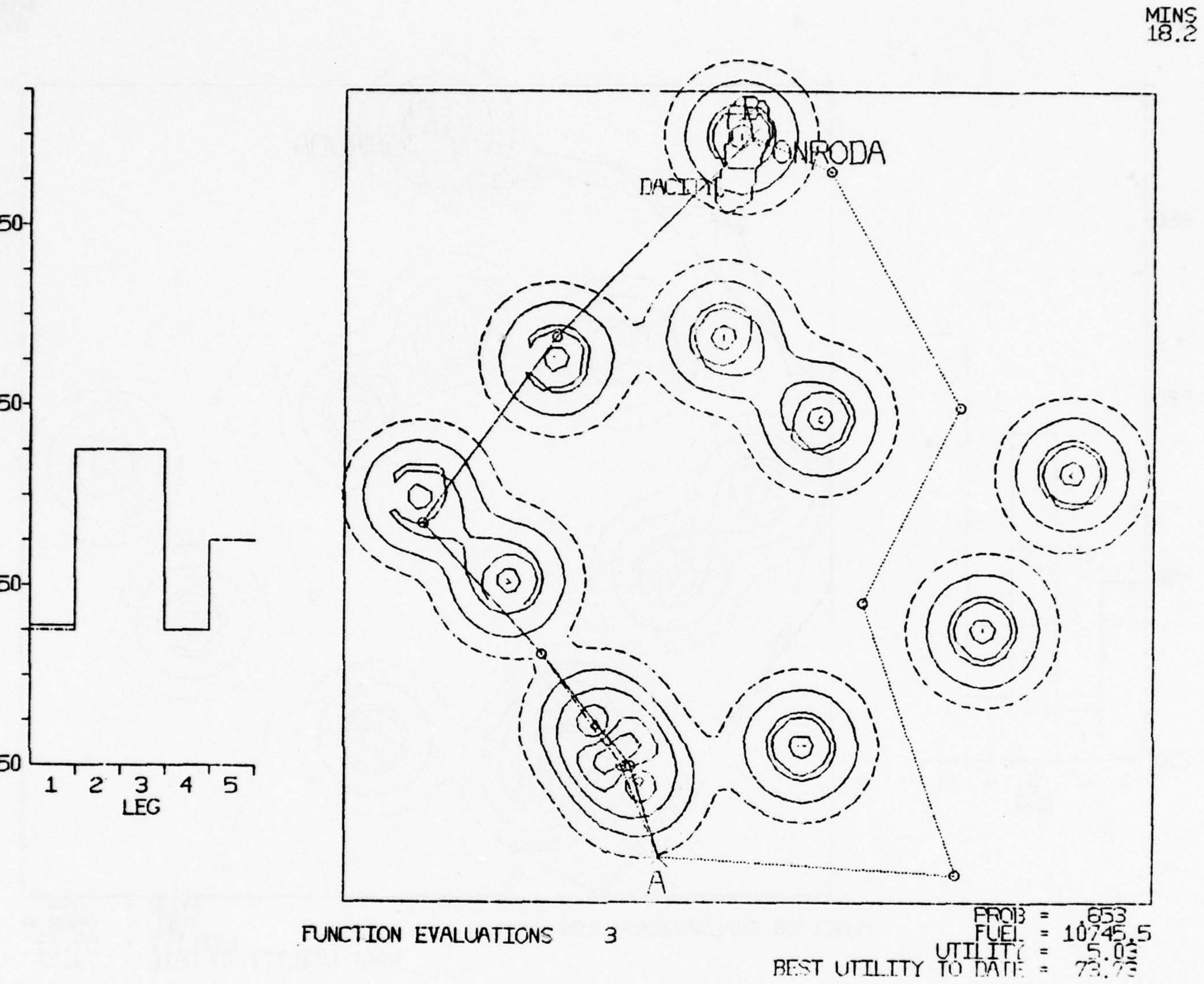
The operator resets the way points to explore the fifth path. At the end of three evaluations the utility is 36.45.

Figure D24. Sixteenth Plate, Example Problem.



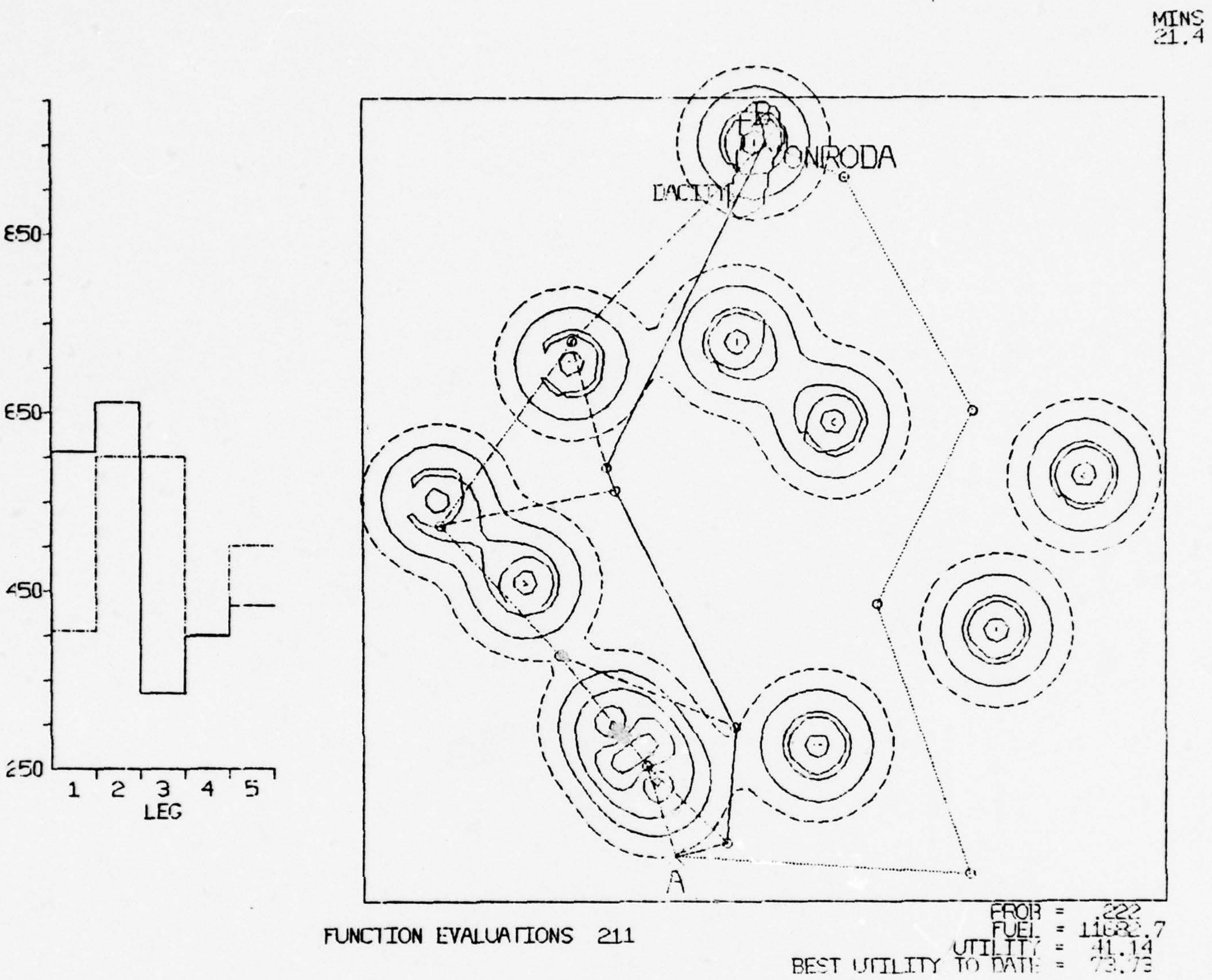
The operator stops the algorithm at the end of 189 evaluations. Note that the algorithm moved the last way point closer to ONRODA. Also, note that the first way point was moved down to get away from the contours above the starting point.

Figure D25. Seventeenth Plate, Example Problem.



The operator chooses a very poor set of way points going through high detection capability contours.

Figure D26. Eighteenth Plate, Example Problem.



At the end of 211 evaluations the algorithm found its way over to the vicinity of the fourth path evaluated. But, clearly, it would never have found its way to the best path found by the operator interacting with the algorithm.

Figure D27. Nineteenth Plate, Example Problem.

APPENDIX E

TRAINING MATERIALS FOR OAO USING DYNAMIC PROGRAMMING ALGORITHM

I. INTRODUCTION

Integrated Sciences Corporation is conducting a study for the Office of Naval Research that investigates ways to allocate functions between humans and computers so that their respective strengths are best used. The portion of the study in which you are participating seeks to determine to what extent, if any, a human operator can aid and thus improve the performance of selected optimization techniques when applied to a naval tactical decision aiding problem. We call this "operator-aided optimization," or OAO for short. The optimization technique you will be working with is dynamic programming optimization. Don't worry if you are unfamiliar with this technique. Even if you have never heard of it, you will learn enough about its characteristics during the training phase to enable you to perform well on the experiment.

Your role in the experiment is to act as the member of a Naval Task Force Commander's (TFC's) staff who is planning a tactical airstrike against the airfield on a place called ONRODA Island. Your Naval Task Force consists of aircraft carriers, their squadrons of aircraft, and escort ships. They are located approximately at the point marked with an X in Figure E1. About ten enemy ships are located in a region between your Task Force and ONRODA. Important parts of air strike planning are (a) deciding the path that the aircraft will take to get to the target and (b) strike aircraft speeds along the legs of the path. As air strike planner, you must be concerned about these two factors:

1. The probability that aircraft will be detected before they reach the target. If they are detected before reaching the target, the enemy will be at maximum readiness to repel the air strike. The enemy ships between your Task Force and ONRODA have radar that could detect your aircraft. However, the enemy ships have no interceptor aircraft nor do they have guns or missiles that would be effective against your aircraft.

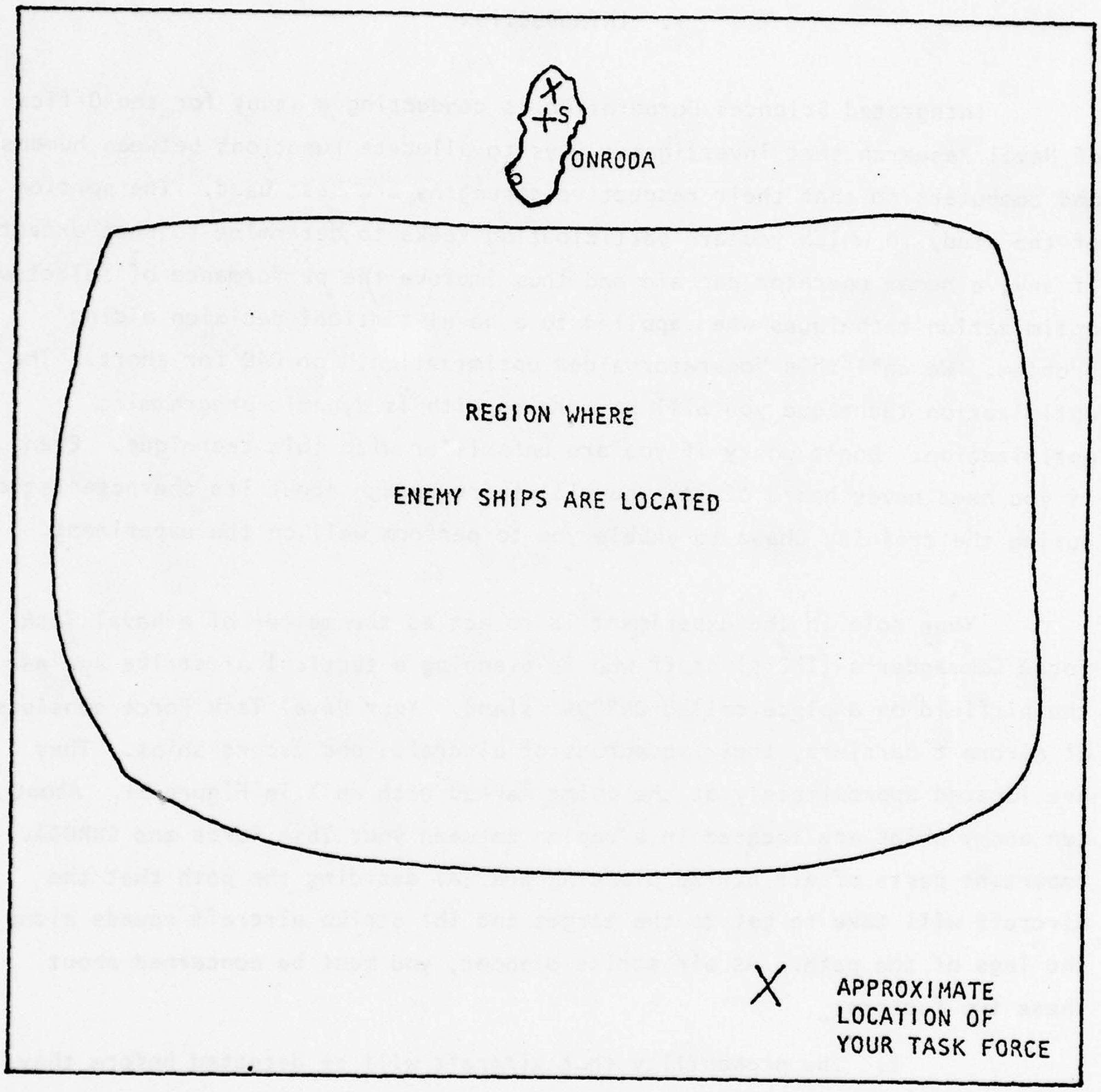


Figure E1. Map of Area of Interest.

2. Amount of fuel left aboard your aircraft when they reach the target. It is desirable to maximize the fuel left in order to engage or avoid enemy interceptor aircraft over the target or to attack secondary targets once the primary target, ONRODA airfield, has been destroyed. Your job is to help the computer come up with the best airstrike plan between the task force and the target within a specified time limit.

The best air strike plan minimizes the probability of the aircraft being detected by the radars and, at the same time, leaves maximum fuel remaining upon arrival at the target so that enemy fighter aircraft can be engaged or evaded.

The purpose of this material is to acquaint you with the:

1. Detection ability of multiple enemy radars when there is overlapping detection coverage between radars in proximity to each other
2. Means of measuring the goodness of an air strike plan
3. Characteristics of the dynamic programming optimization technique.

The training goals are to:

1. Develop expertise in using the equipment
2. Develop a feel for the best way to help the computerized technique find the best air strike paths and speeds.

In training you will do eight problems with the optimization technique. Experimental data collection will then be done for twelve problems. Thus, you will do a total of 20 problems. Each problem will last 15 minutes.

A. REPRESENTATION OF ENEMY RADAR DETECTION CAPABILITY

The capability of a single enemy radar to detect your aircraft is represented by concentric circles around the radar location. Detection capability is the same at all points on each circle and is a specified percentage of the peak detection capability of the radar. (See Figure E2.) Notice that as you go along a radial line toward the center of the concentric circles, detection capability increases up to the 90% of the peak level. The peak occurs between the two 90% circles and detection capability decreases from the peak as you get closer to the radar location. Thus detection capability may be visualized in three dimensions as a volcano with a rim and a crater in the center of the volcano. The "Detection volcano" is centered on the radar's position.

When several radars have overlapping coverage as shown in Figure E3, the probability of detecting your aircraft at a point within areas of overlap is higher than it would be at that same point if only one radar could detect at that point. Thus there is a joint detection capability throughout areas of overlap. The points where joint probabilities of detection are equal are connected together to form contours as shown in Figure E4. The contours have the same general meaning as the concentric circles in Figure E2, that is, each contour is the set of points where detection capability is some specified percentage of the peak joint detection capability. The set of contours is analogous to a topographical map. The difference is that each contour on a topographical map corresponds to an altitude above sea level and each detection capability contour corresponds to a detection capability between zero capability and the peak capability.

B. MEASURING THE GOODNESS OF AN AIR STRIKE PATH: THE UTILITY FUNCTION

The problem is to select an optimal air strike path, so an appropriate utility criterion function is one which measures the "goodness" of an air strike path. The two variables selected to determine the goodness of an air strike path were fuel consumption along the path and probability of being

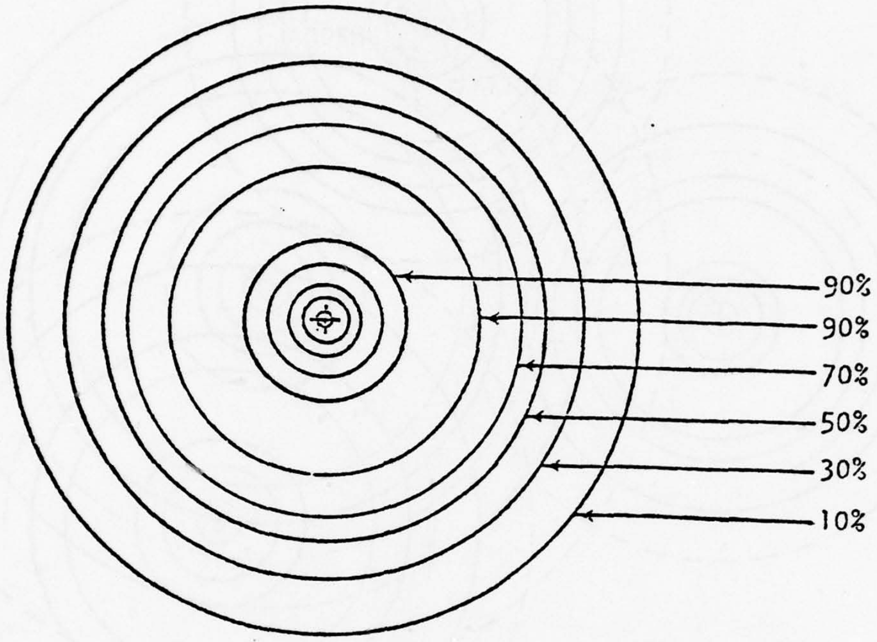


Figure E2. Single Sensor Coverage Template. (Circles show distance from sensor location, center, at which percentage values of the peak detection rate occur.)

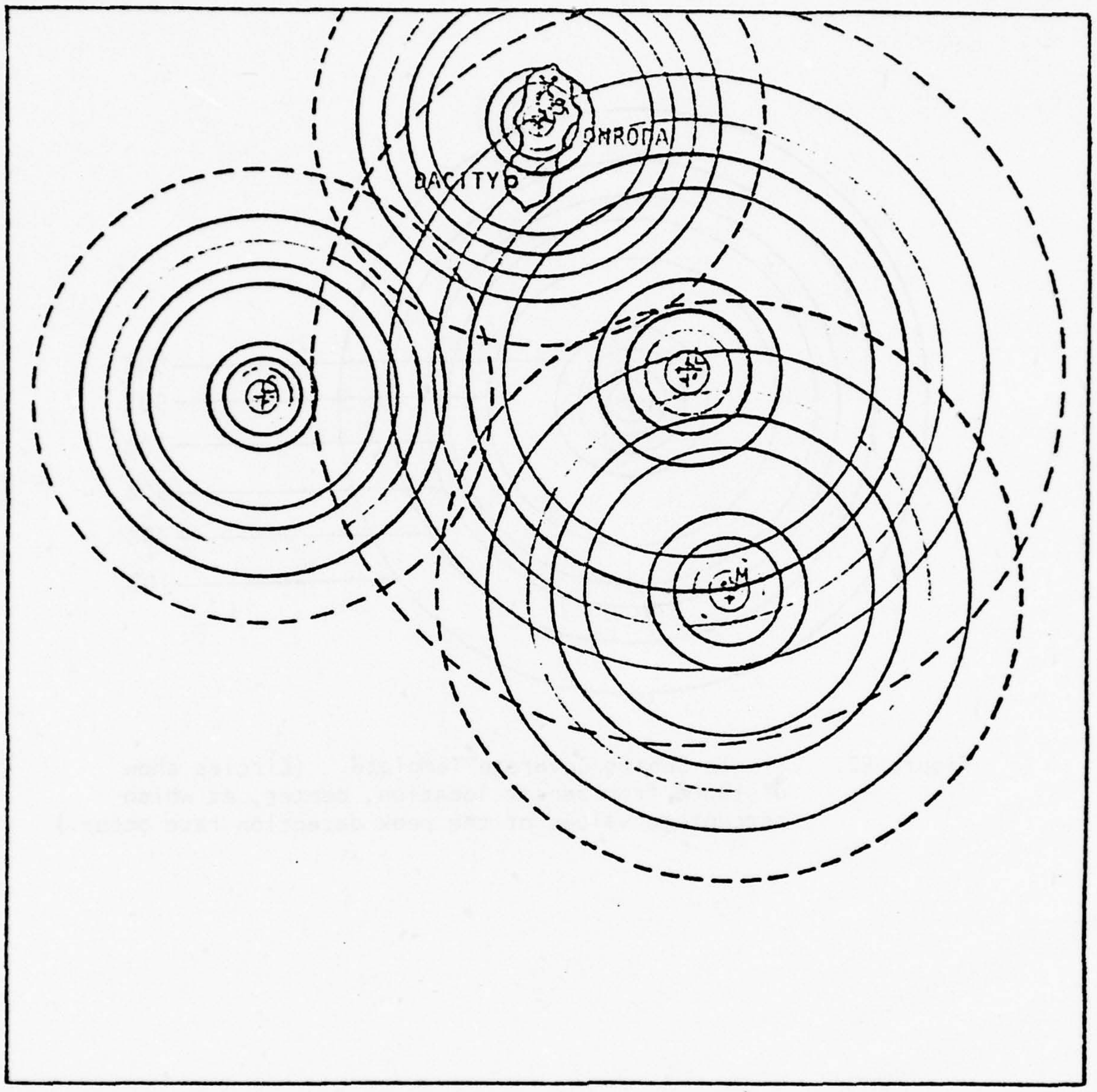


Figure E3. Display with all Sensor Coverage Templates Shown.

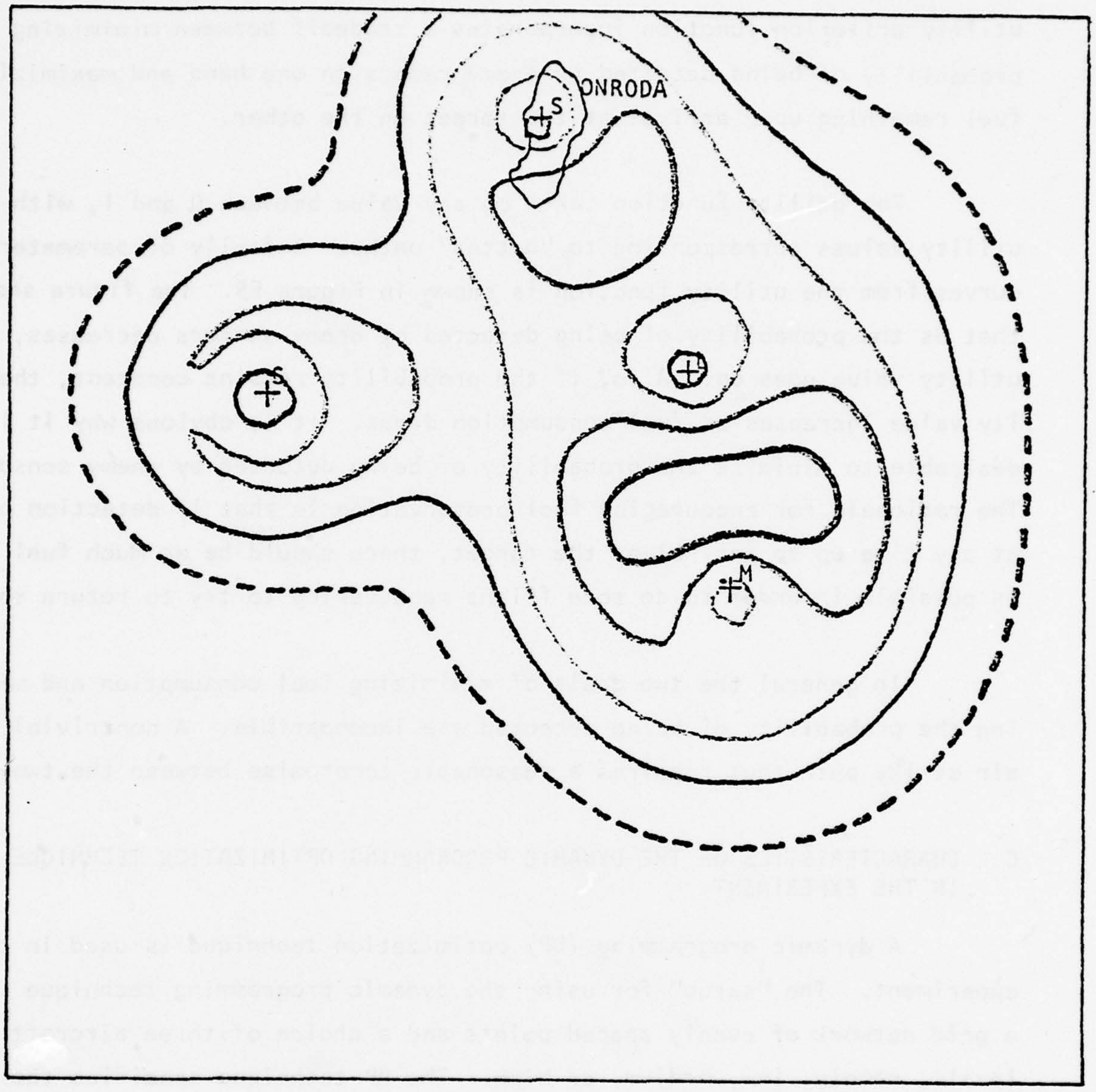


Figure E4. Contours Showing Joint Probabilities of Detection for Four Radars Located at Positions Marked with a Cross (+).

detected by one or more enemy radars prior to reaching the target. The utility criterion function incorporates a tradeoff between minimizing the probability of being detected by enemy radars on one hand and maximizing the fuel remaining upon arrival at the target on the other.

The utility function takes on any value between 0 and 1, with higher utility values corresponding to "better" paths. A family of parameterized curves from the utility function is shown in Figure E5. The figure shows that as the probability of being detected by enemy sensors decreases, the utility value goes up. Also, if the probability remains constant, the utility value increases as fuel consumption drops. It is obvious why it is desirable to minimize the probability of being detected by enemy sensors. The rationale for encouraging fuel preservation is that if detection occurs at any time up to arrival at the target, there should be as much fuel left as possible in order to do some flight maneuvering to try to return safely.

In general the two goals of minimizing fuel consumption and minimizing the probability of being detected are incompatible. A nontrivial optimal air strike path thus requires a reasonable compromise between the two goals.

C. CHARACTERISTICS OF THE DYNAMIC PROGRAMMING OPTIMIZATION TECHNIQUE USED IN THE EXPERIMENT

A dynamic programming (DP) optimization technique is used in the experiment. The "setup" for using the dynamic programming technique includes a grid network of evenly spaced points and a choice of three aircraft speed levels, namely, low, medium, or high. The DP technique specifies the best path by connecting points on the grid between air strike launch point and target and specifying one of the three speeds for each path leg between two connected points. The advantage of DP is that it does find the best path for the grid and speed levels it is using. The disadvantage of DP is that it takes a long time (even with the computer doing the number crunching) to

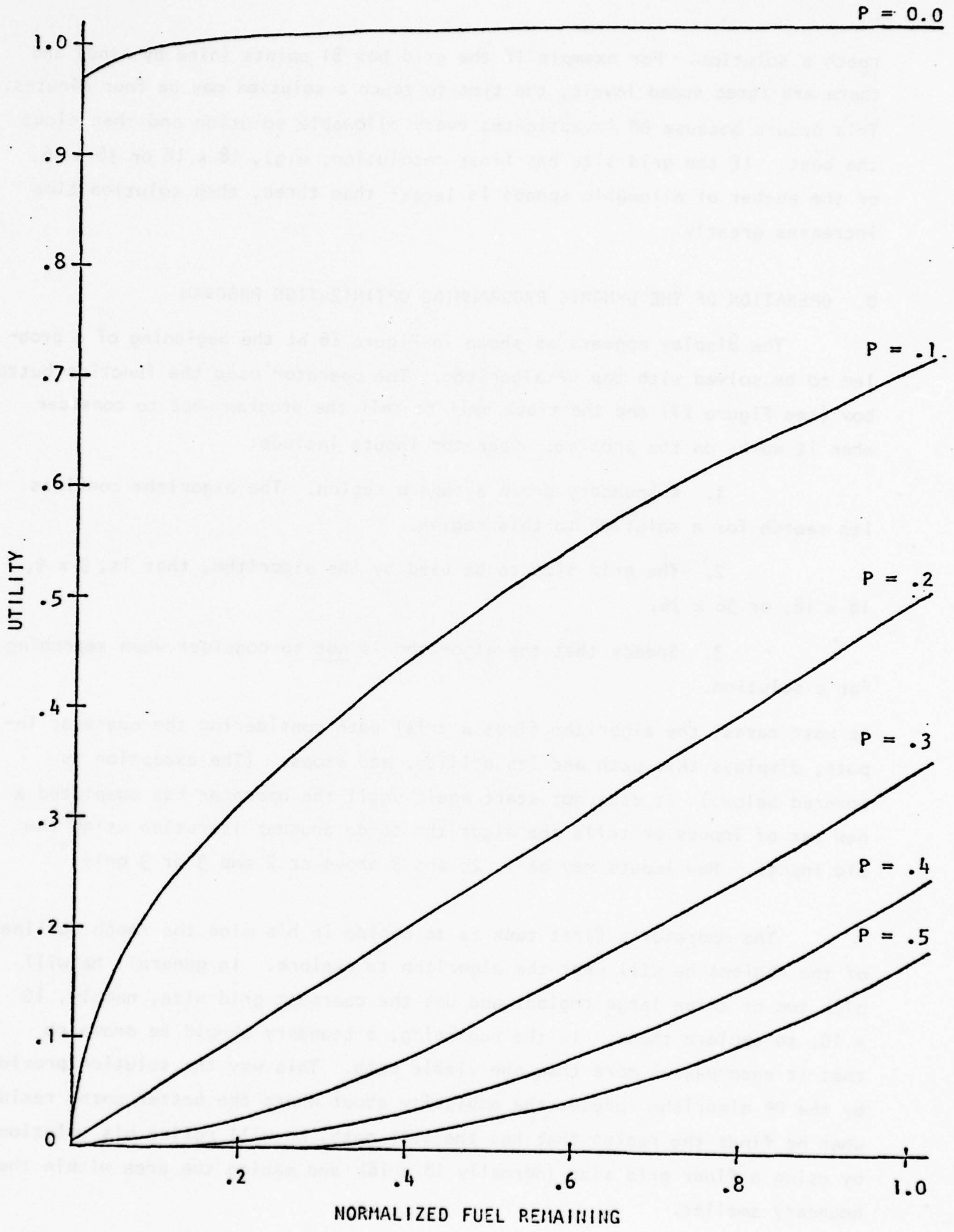


Figure E5. Family of Parameterized Utility Function Curves.

reach a solution. For example if the grid has 81 points (nine by nine) and there are three speed levels, the time to reach a solution may be four minutes. This occurs because DP investigates every allowable solution and then picks the best. If the grid size has finer resolution, e.g., 18 x 18 or 36 x 36, or the number of allowable speeds is larger than three, then solution time increases greatly.

D. OPERATION OF THE DYNAMIC PROGRAMMING OPTIMIZATION PROGRAM

The display appears as shown in Figure E6 at the beginning of a problem to be solved with the DP algorithm. The operator uses the function button box (see Figure E7) and the track ball to tell the program what to consider when it works on the problem. Operator inputs include:

1. A boundary drawn around a region. The algorithm confines its search for a solution to this region.

2. The grid size to be used by the algorithm, that is, 9 x 9, 18 x 18, or 36 x 36.

3. Speeds that the algorithm is <u>not</u> to consider when searching for a solution.

In most cases, the algorithm finds a trial path considering the operator inputs, displays this path and its utility, and stops. (The exception is covered below.) It does not start again until the operator has completed a new set of inputs or tells the algorithm to do another iteration using the old inputs. New inputs may be 1, 2, and 3 above or 2 and 3 or 3 only.

The operator's first task is to decide in his mind the rough outlines of the regions he will want the algorithm to explore. In general, he will pick two or three large regions and use the coarsest grid size, namely, 10 x 10, to explore these. In the beginning, a boundary should be drawn so that it encompasses more than one viable path. This way the solution provided by the DP algorithm reduces the ambiguity about where the better paths reside. When he finds the region that has the best path, he will refine his solution by using a finer grid size (normally 18 x 18) and making the area within the boundary smaller.

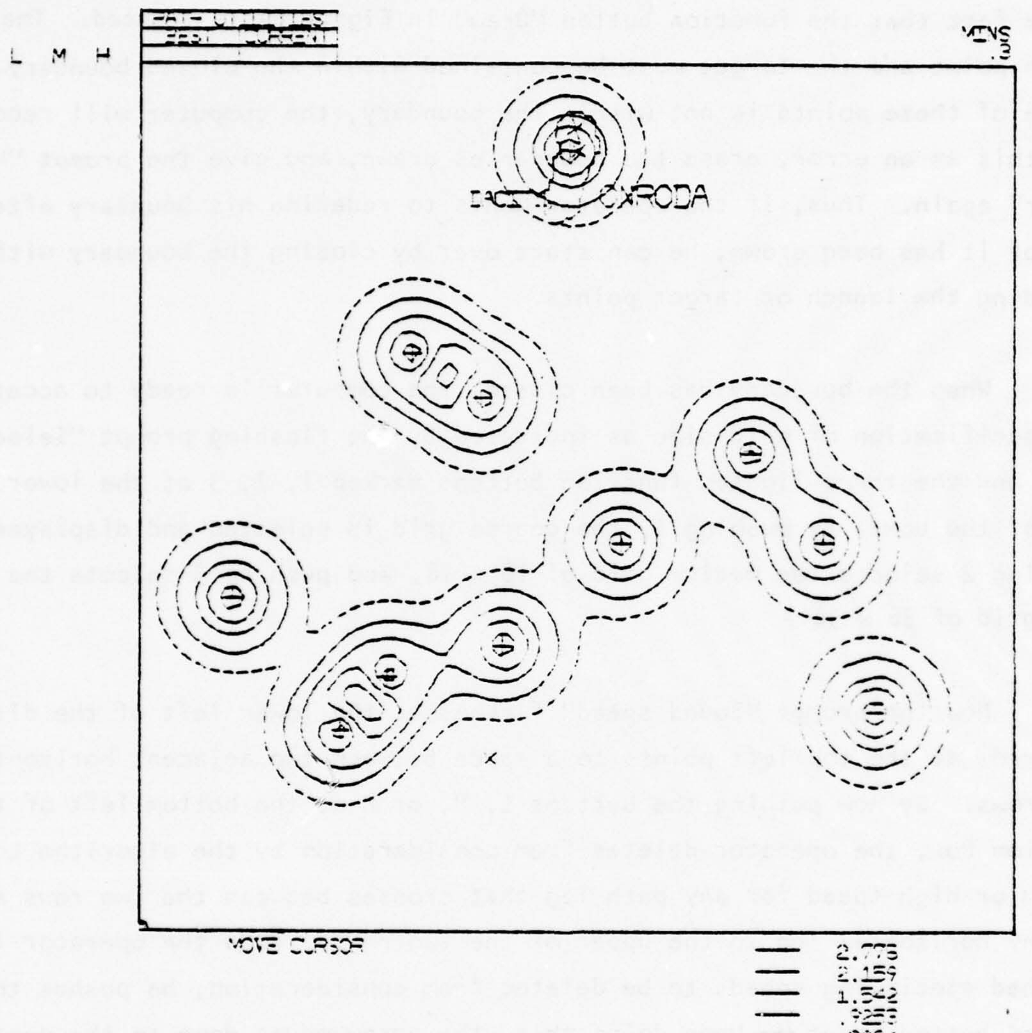


Figure E6. Display Appearance at Beginning of Problem to be Solved Using DP Algorithm.

When the operator has decided the first region he wants to explore, he responds to the flashing prompt "Move cursor" at the lower left of the display by moving the track ball to draw the boundary. This is indicated by the fact that the function button "Draw" in Figure E7 is lighted. The launch point and the target must be contained within the closed boundary. If one of these points is not within the boundary, the computer will recognize this as an error, erase the boundaries drawn, and give the prompt "Move cursor" again. Thus, if the operator wants to redefine his boundary after some of it has been drawn, he can start over by closing the boundary without including the launch or target points.

When the boundary has been closed, the computer is ready to accept the specification of grid size as indicated by the flashing prompt "Select grid" and the three lighted function buttons marked 1, 2, 3 at the lower left of the box. By pushing 1, the coarse grid is selected and displayed. (Pushing 2 selects the medium grid of 18 x 18, and pushing 3 selects the fine grid of 36 x 36.)

Now the prompt "Bound speed" flashes at the lower left of the display. The arrow at the top left points to a space between two adjacent horizontal grid rows. By now pushing the buttons L, M, or H at the bottom left of the function box, the operator deletes from consideration by the algorithm Low, Medium or High speed for any path leg that crosses between the two rows and for any horizontal leg in the upper of the two rows. When the operator is finished specifying speeds to be deleted from consideration, he pushes the lighted button "Next." Upon doing this, the arrow moves down to the next pair of rows and the operator repeats the procedure. For example:

1. If the operator has pushed L, M, and Next, the algorithm will not consider low and medium speeds for any leg crossing between the two rows on either side of the arrow and for any horizontal leg in the upper row.

2. If the operator has pushed only "Next," then the algorithm will not delete any speeds and the arrow will move down to the next pair of rows.

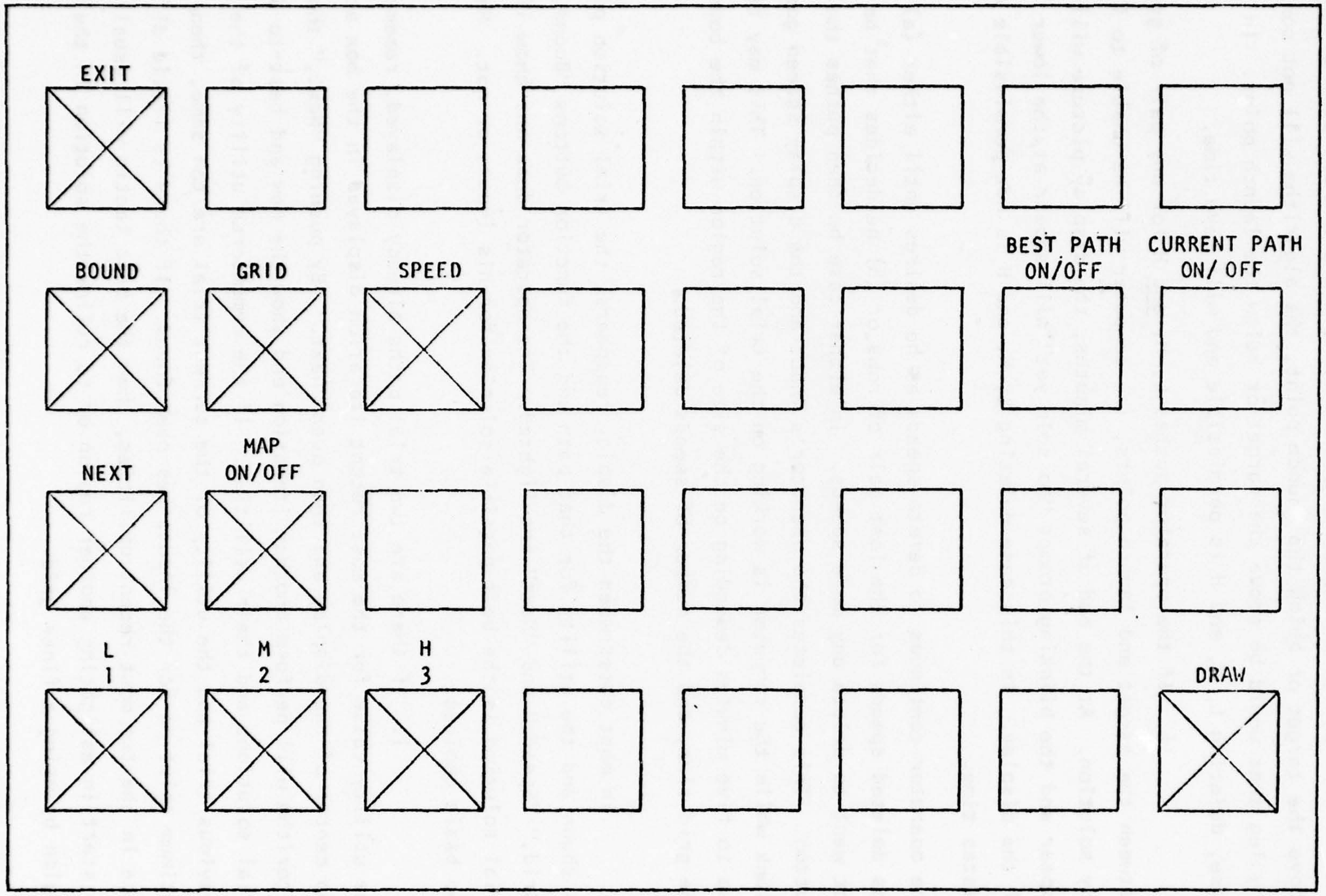


Figure E7. Function Button Configuration for the Dynamic Programming Optimization.

3. If the operator pushes L, M, <u>and</u> H for any pair of grid rows above the target or below the launch point, the algorithm will not consider any leg that would be above the target or below the launch point. In this case, deleting L, M, and H is permissible and will save time.

4. If the operator pushes L, M, <u>and</u> H for any pair of grid rows between the target and launch points, the computer will be unable to find any solution. At the end of several minutes, the display picture will reappear and the blinking prompt "No soln yet" will appear at the lower left of the display. In this case deleting L, M, and H is not permissible and wastes time.

The operator continues to delete speeds as he desires until either (a) he has deleted speeds for the last pair of rows or (b) he decides that he does not want to delete any more speeds. In either case he then pushes the Exit button. This completes the operator's input and the display screen goes blank while the computer is working on the trial solution. This may take two to five minutes depending on the size of the region within the boundary, the grid size, and the number of speeds deleted.

In most cases when the display reappears, the trial solution path is shown and the utility for that path and the function buttons "Bound," "Grid," "Speed," and "Next" are lighted. The operator does not know if the trial solution is the best possible solution for his inputs or not. He has two basic choices:

1. If there are two trial paths already displayed, remember the utility value for the most recent iteration displayed in the box at the top center of the display and then push "Next." By pushing "Next," the algorithm will perform another iteration and show the new and best-to-date trial solutions and their utilities. If the remembered utility of the previous trial and the utility of the current trial are the same, then the optimum solution for the inputs has been found. If there is little difference in the two most recent utilities, then the best tactic will usually be to start investigating another region or to refine the solution in the current region by using a finer grid.

2. By pushing Bound, Grid, or Speed he can redefine the inputs considered by the algorithm. If he pushes Bound, then he must go through all three steps of drawing the boundary, selecting grid size, and deleting speeds. If he pushes Grid, then the algorithm will use the previously drawn boundary and the operator selects grid size and deletes speeds. If he pushes Speed, the algorithm accepts the previously drawn boundary and grid and the operator only deletes speeds. If the area within the boundary was large, then the operator should redefine the boundary to include a much smaller number of points in the vicinity of the path selected by the previous iteration.

If the display reappears without a new trial solution (the exception previously noted), "No Soln Yet" will flash at the lower left of the display. This means that the algorithm has not been able to find a complete trial solution on a single iteration. In this case the operator's suggested response is to push the Next button so that the algorithm will go to the next iteration to complete the trial solution.

E. GUIDELINES

There are two types of data being analyzed:

1. Utility of the operator's first best estimate of the best path and leg speeds. The operator will draw his first best estimate on a hard copy plot of the problem and write the letter symbols for each leg's speed. This will be done prior to the operator beginning use of the display. However, problem time will start as soon as the operator receives the hard copy plot of the problem. The operator's first goal is to do the best he can on his "first best estimate."

2. Operator performance will be calculated at the end of each trial by adding the 15 utilities of the best-solution-to-date at the end of each minute and dividing this sum by 15. Thus, operator performance for the entire trial is the average of the 15 utilities. The operator's second goal is to maximize this average.

In general, (2) above is done by exploring the regions which could contain the best path in the order of estimated likelihood that each contains the best path. If the operator is correct concerning the region which contains the best path, then the average utility will be nearly equal to the utility of the best path. This is true because the computer only stores the best utility to date and will therefore not store the utilities of paths investigated after the first when the first region explored contains the best path.

The problem is deciding how large a region should be explored at first. Since the viable paths will certainly not cross each other, they can be represented as a left-to-right sequence as shown below for an assumed five paths.

Paths	→	A	B	C	D	E
Operator estimate if likelihood of best path: Case I	→	4	3	5	1	2
Case II	→	1	4	3	5	2
Case III	→	5	1	2	4	3
Case IV	→	5	1	4	2	3

In Case I it would probably be best to draw a boundary around D and E for the first trial. In Case II it's a tossup whether to look at A alone or A and B simultaneously. Looking at A alone will save time and therefore get you "on the board" soonest. However, if B is actually best, you won't find it until after you've looked at D and E together. You should look at B and C together, then D and E in Case III. Look at A only if the previous results leave you dubious about your initial estimate of A. In Case IV you may want to explore A, B, and C simultaneously, then D and E. Only in Case I is the choice rather clear. In all the others it's a highly subjective matter.

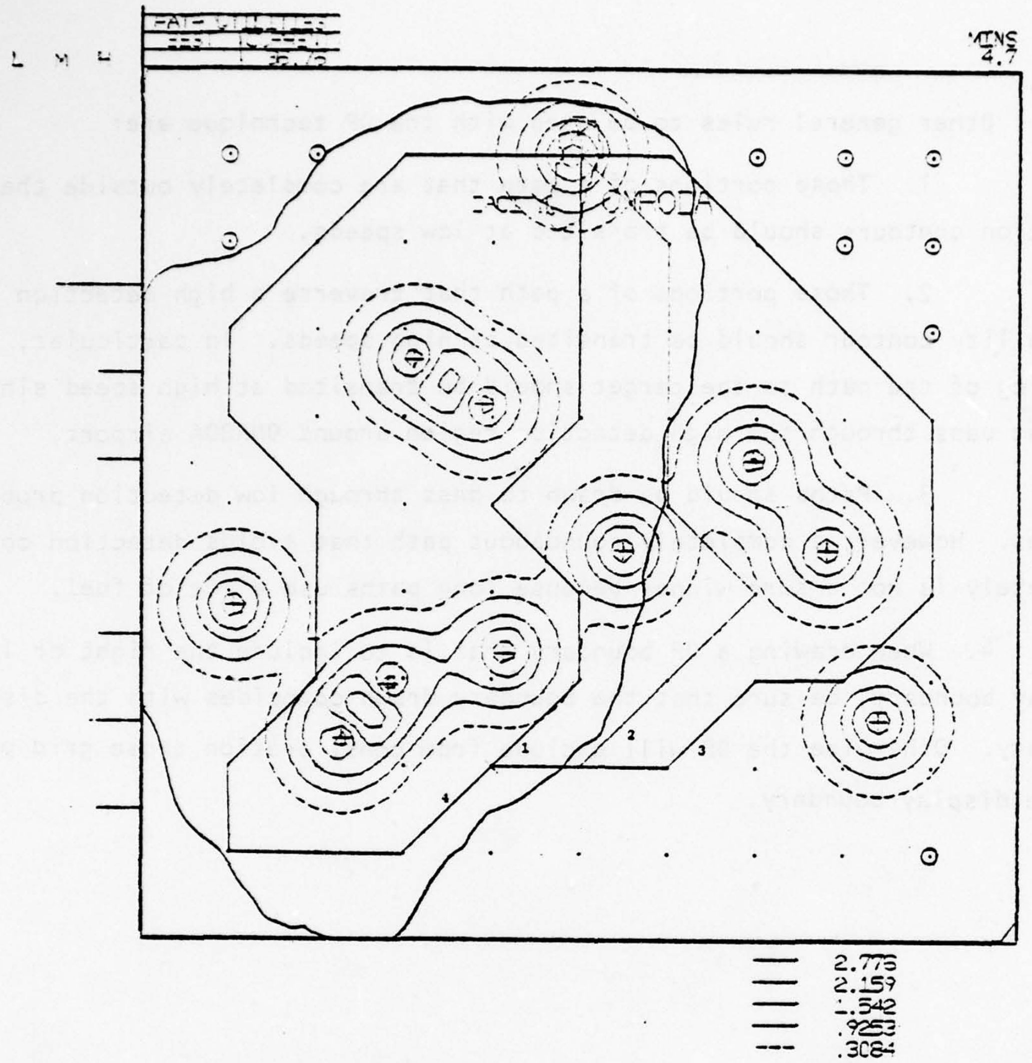
Other general rules to be used with the DP technique are:

1. Those portions of a path that are completely outside the detection contours should be transited at low speeds.

2. Those portions of a path that traverse a high detection probability contour should be transited at high speeds. In particular, the last leg of the path to the target should be transited at high speed since it must pass through the high detection region around ONRODA airport.

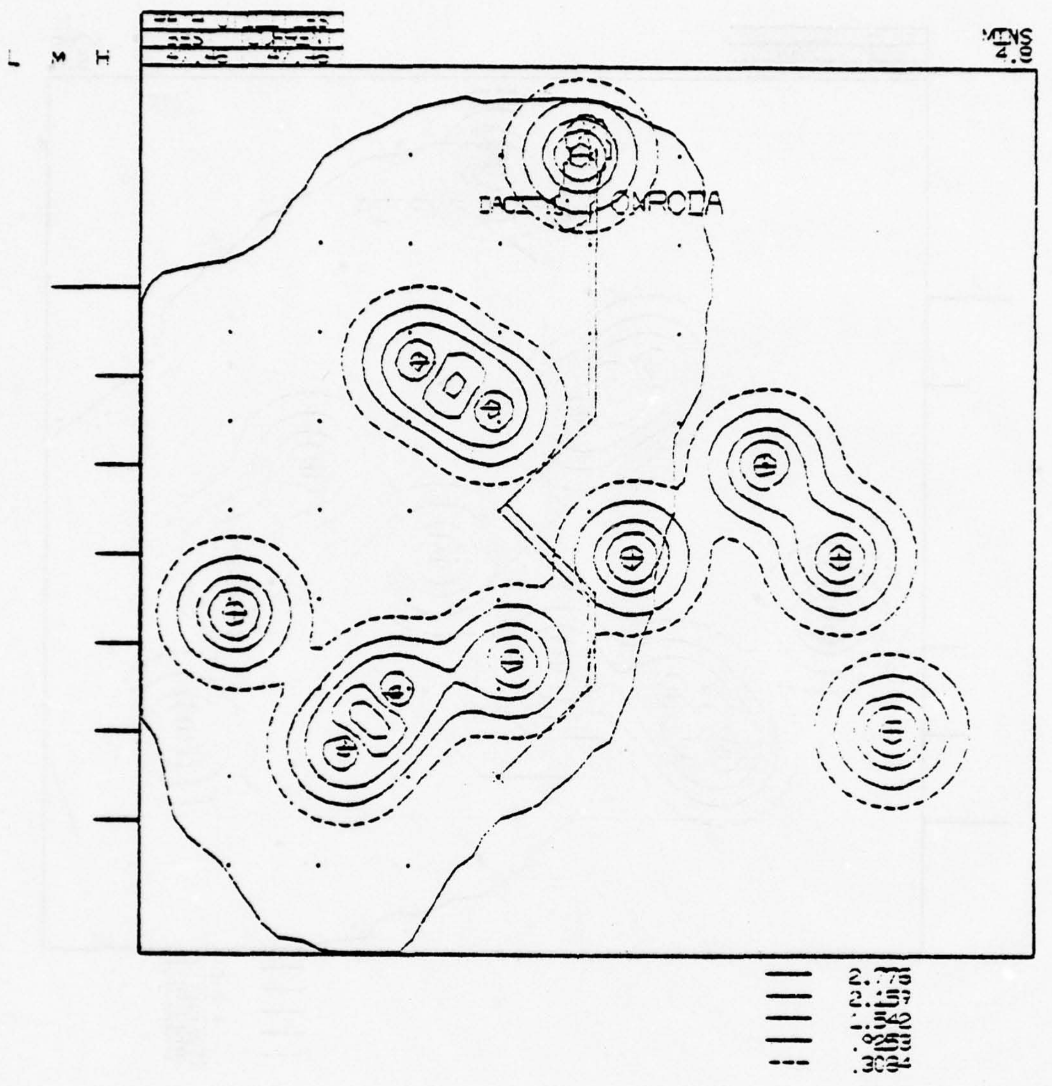
3. Paths should be drawn to pass through low detection probability regions. However, a completely roundabout path that avoids detection contours completely is not a sure winner because long paths use a lot of fuel.

4. When drawing a DP boundary that is to include the right or left display boundary, be sure that the boundary drawn coincides with the display boundary. Otherwise the DP will exclude from consideration those grid points on the display boundary.



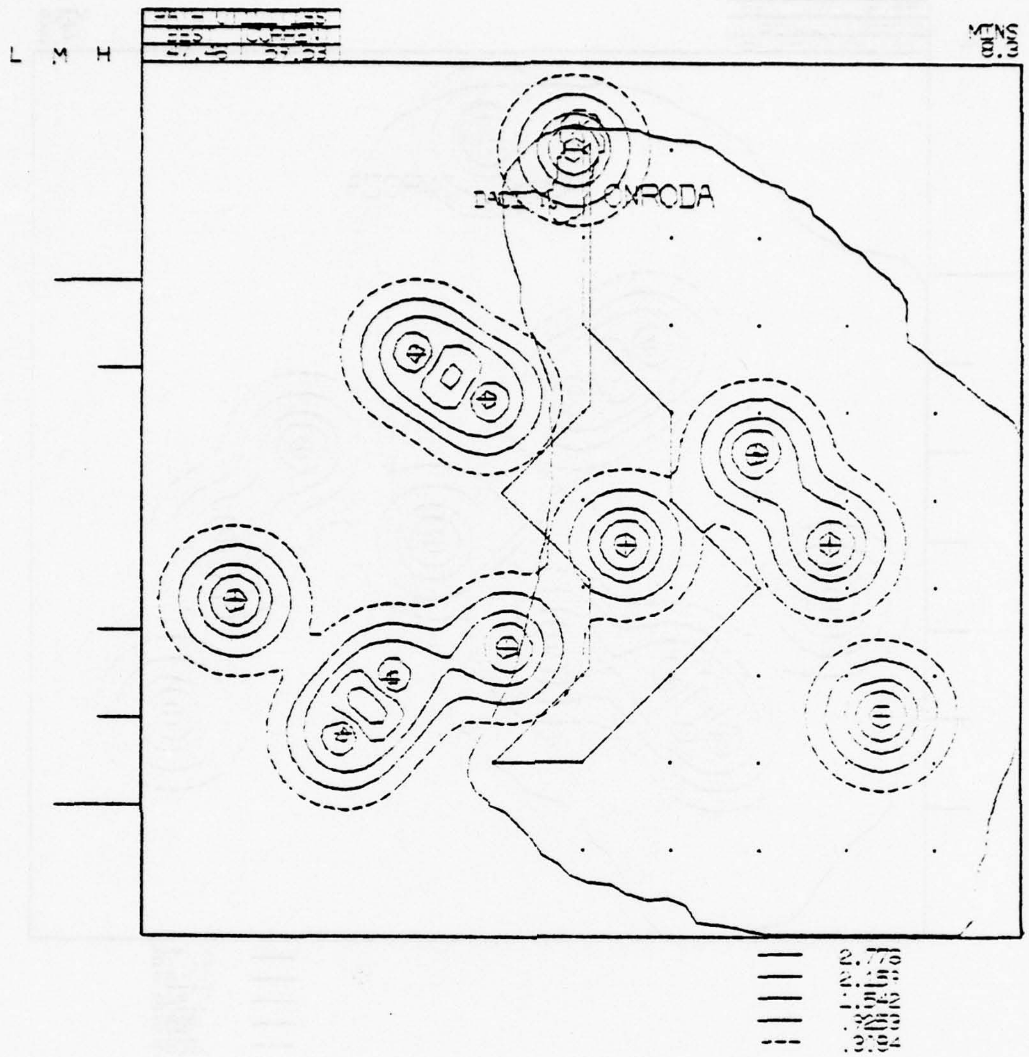
At the beginning of the problem the operator should picture to himself the reasonable candidates for best path. In the figure these candidates are numbered 1-4. Then he should draw a boundary that includes about half of the candidates. Notice that the boundary drawn crosses the midpoint of the display on the right. The reason is that a candidate path that lies mainly in the left half may cross the middle for part of its length. Circled points should not be included within any boundary because the best solution will certainly never include these points. Keeping these points outside the boundary shortens running time for the algorithm. No speeds have been eliminated for the transitions between the two highest rows because low speed will probably be used for one leg of path #4 and high speed will probably be used for the last leg to the target. High speed has been eliminated for most of the remaining transitions because the operator believed high speed would not be selected by the algorithm for the remaining legs on paths #1 and #4.

Figure E8. First Plate, Example Problem.



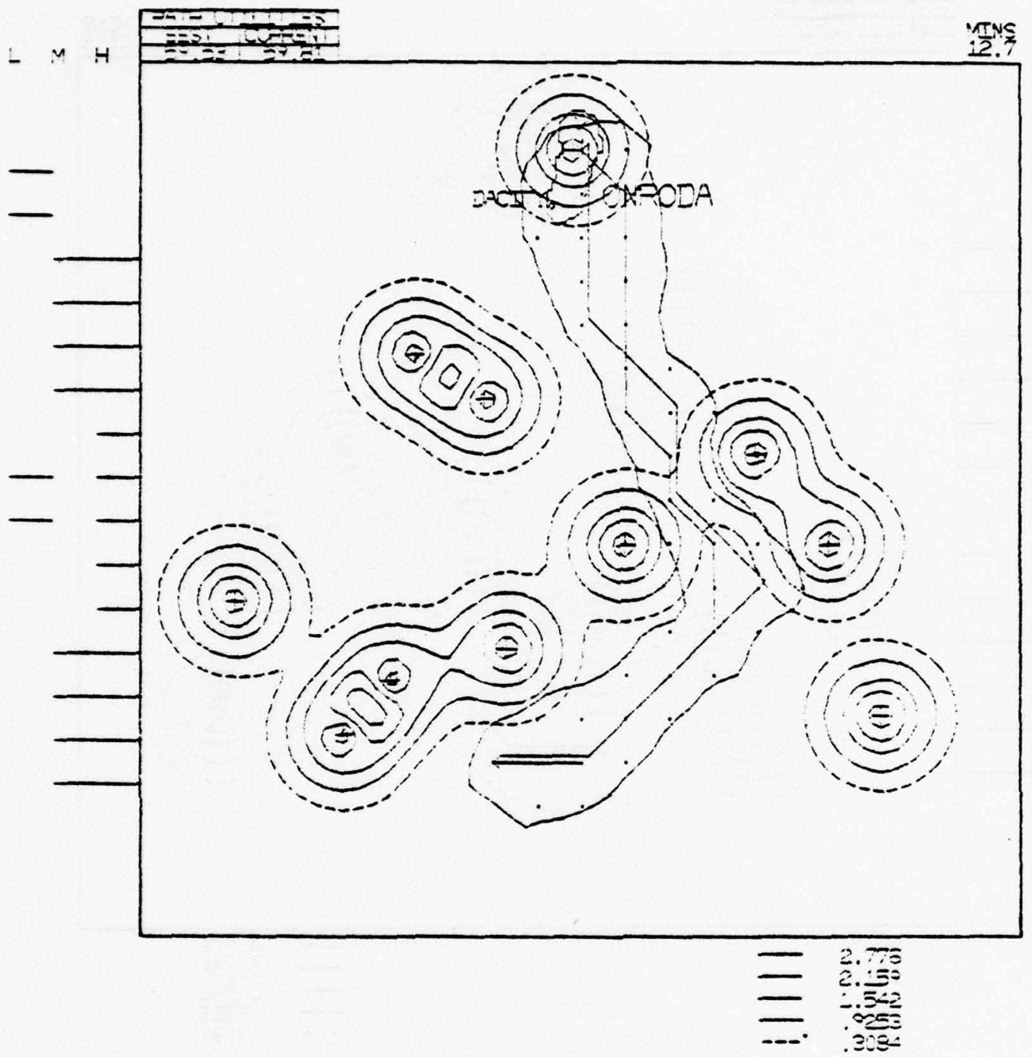
At the end of one iteration for the set of constraints shown in Figure E8, the operator pushes "Next" and finds that another iteration causes no improvement in results.

Figure E9. Second Plate, Example Problem.



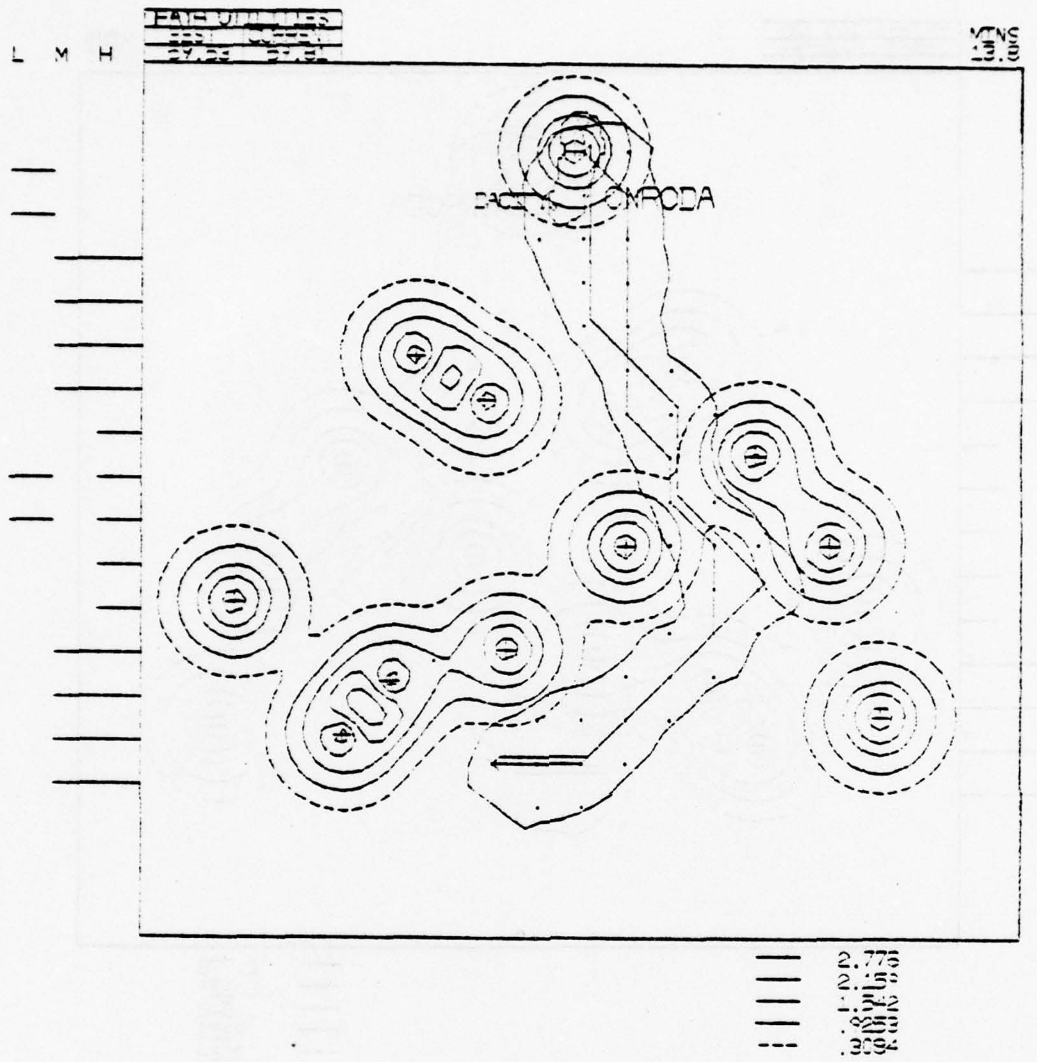
The operator now draws a boundary that will enable the algorithm to consider paths #2 and #3. Note that the boundary does not include the points circled in Figure 1. Also, the operator has placed no speed constraints on transitions in the middle of the display. The reason is that, if the algorithm selects path #2, high or medium speed may be preferable to low speed. On the other hand, if the algorithm selects path #3, low speed will be preferred. The path solved by the algorithm using the operator-supplied constraints resembles path #2 in Figure E8.

Figure E10. Third Plate, Example Problem.



Since the previous solution was the best to date, the operator now decides to refine the solution by using a finer grid size. The result of the new inputs is the path which connects points while the previous best path is shown slightly offset from grid points.

Figure E11. Fourth Plate, Example Problem.

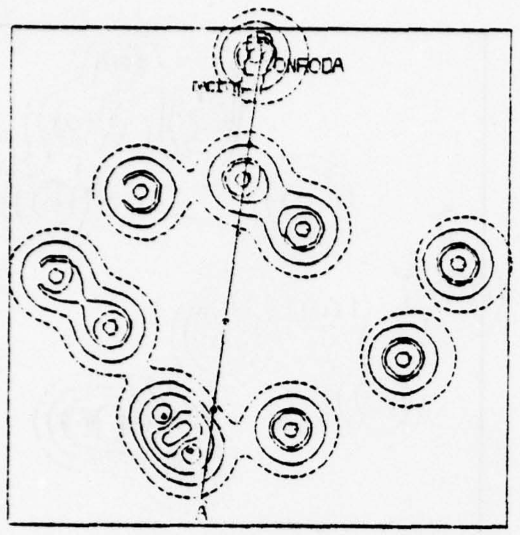


The operator then pushed "Next" and, at the end of another 1.1 minutes, the result was the same shown for the previous iteration.

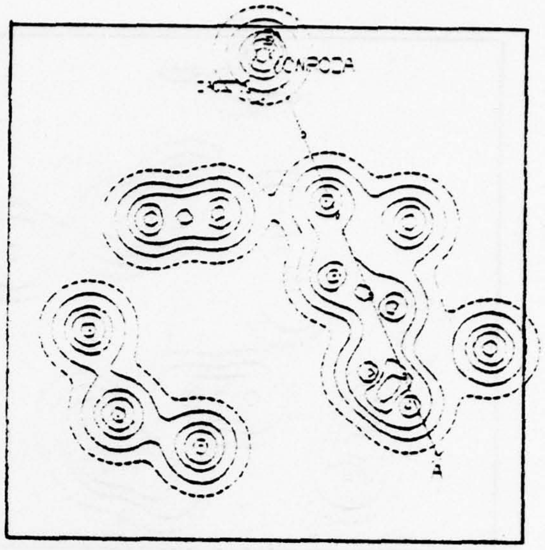
Figure E12. Fifth Plate, Example Problem.

APPENDIX F

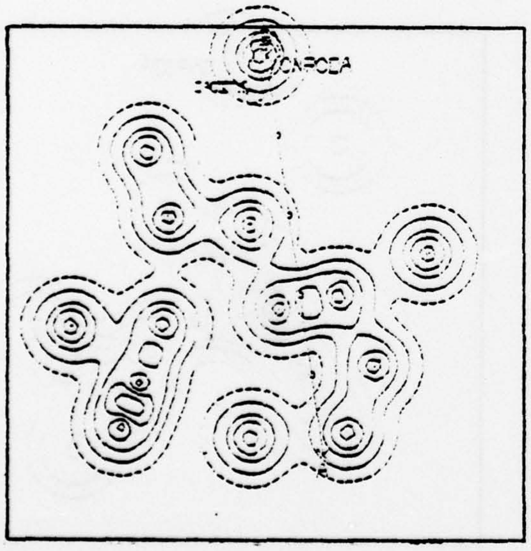
COMPOSITE DETECTION RATE CONTOURS



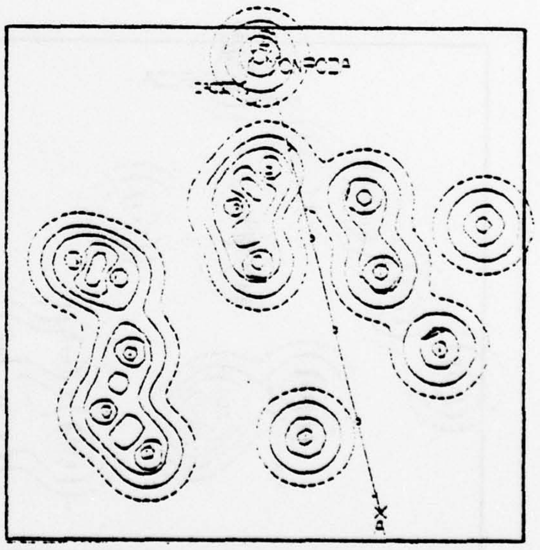
NP Problem 1



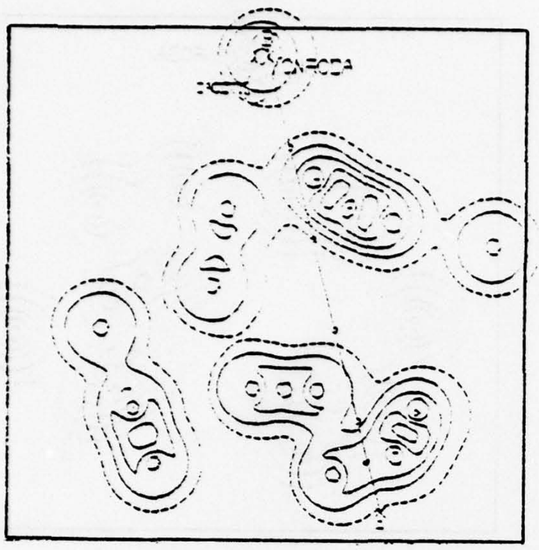
NP Problem 2

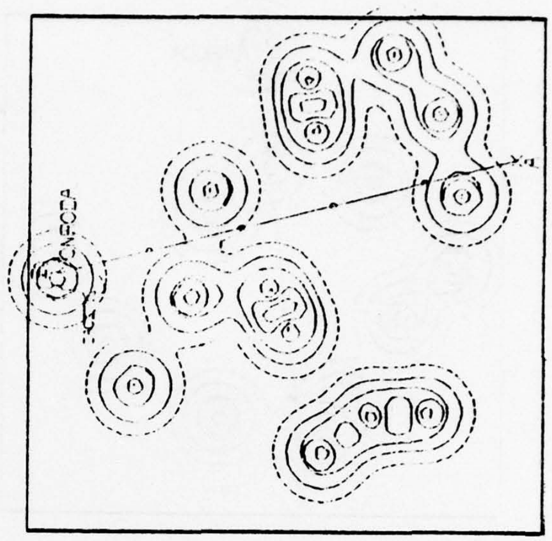


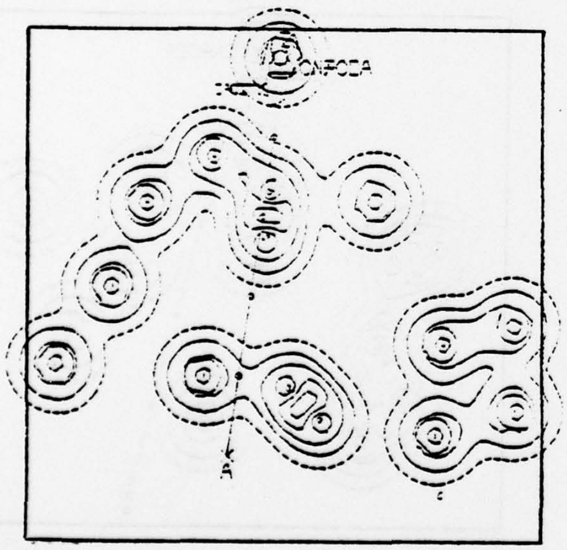
NP Problem 3

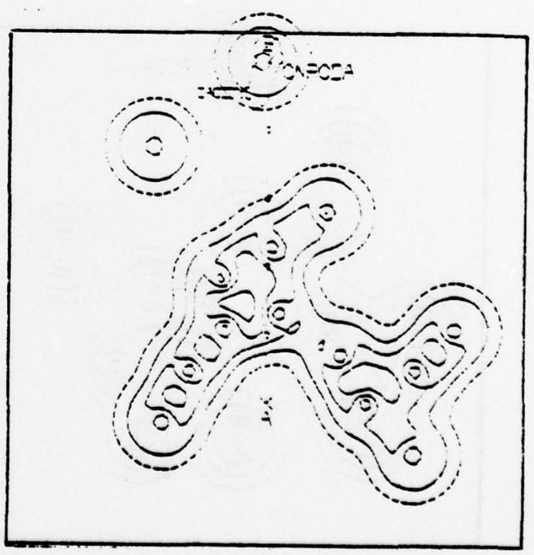


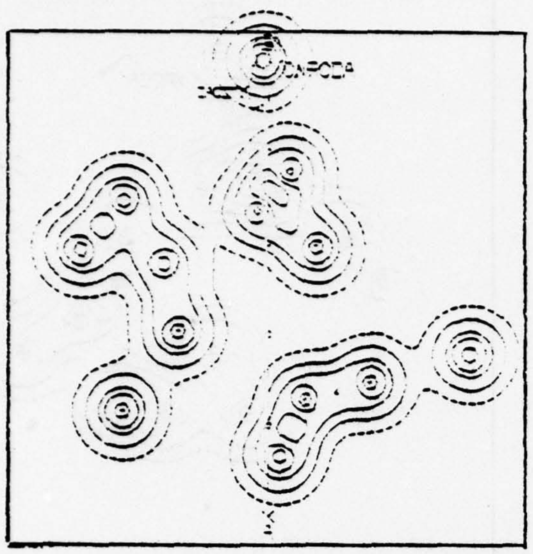
NP Problem 4

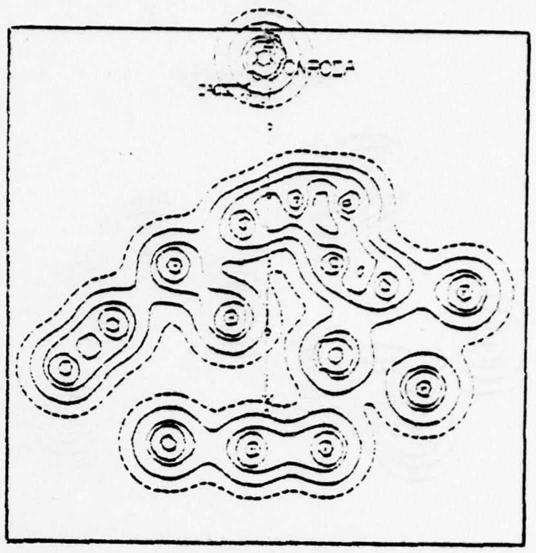

NP Problem 5

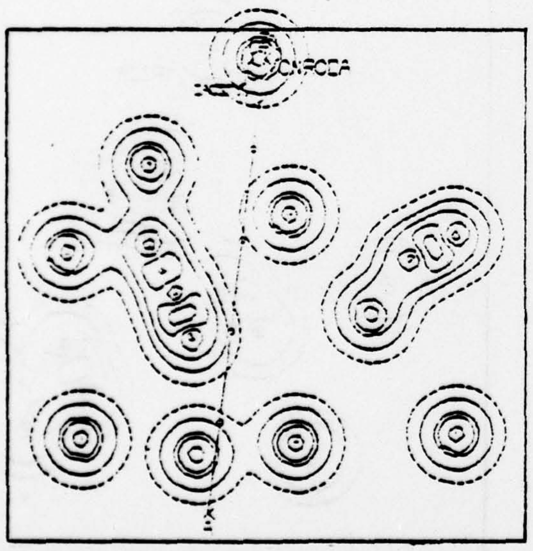

NP Problem 6

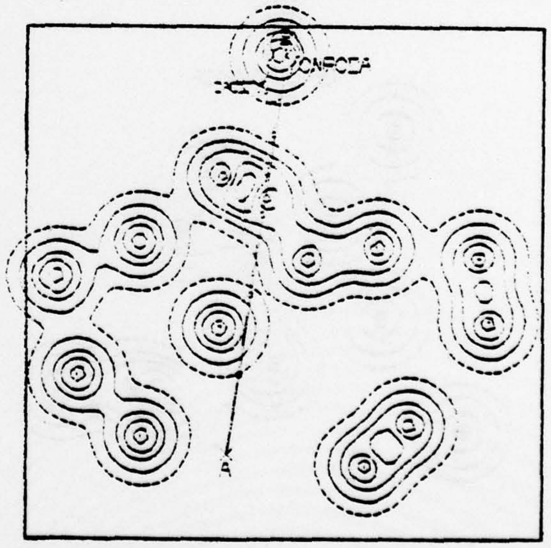

NP Problem 7

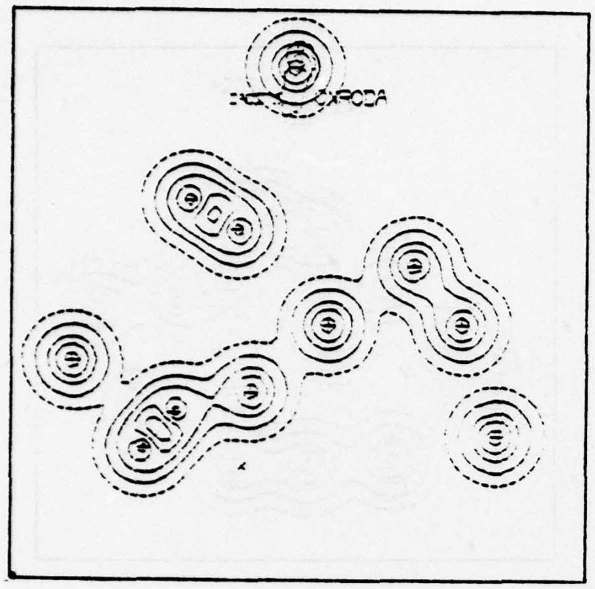

NP Problem 8


NP Problem 9

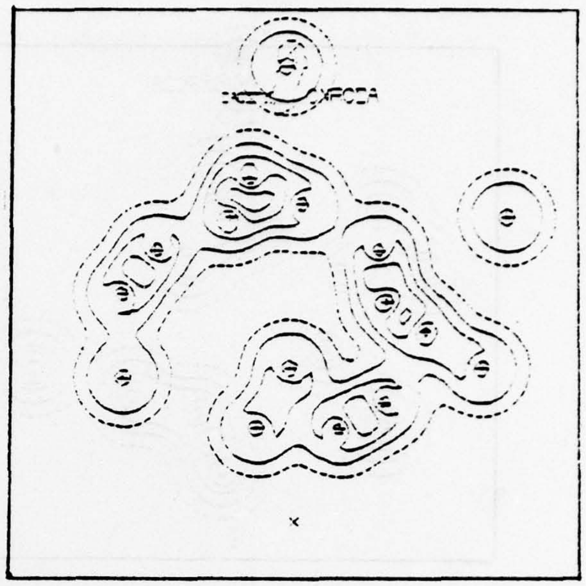

NP Problem 10


NP Problem 11

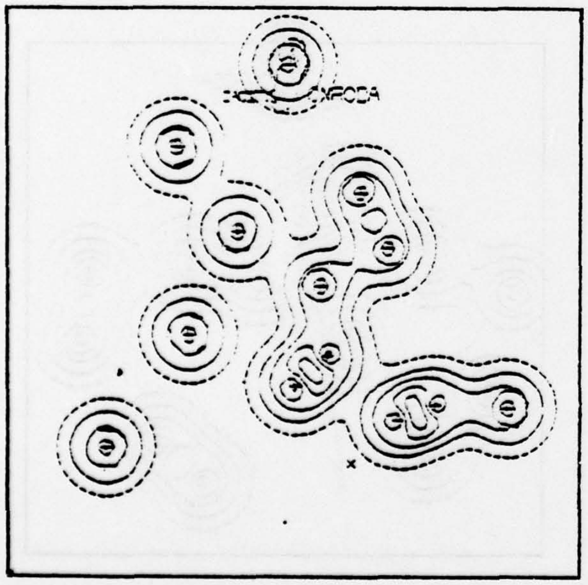

NP Problem 12



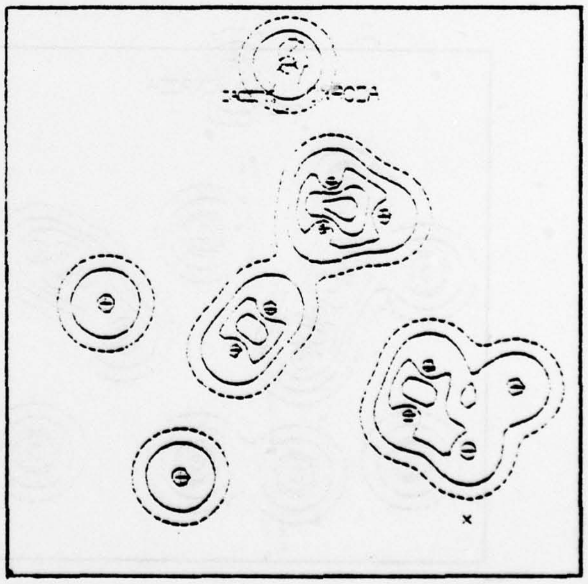
DP Problem 1



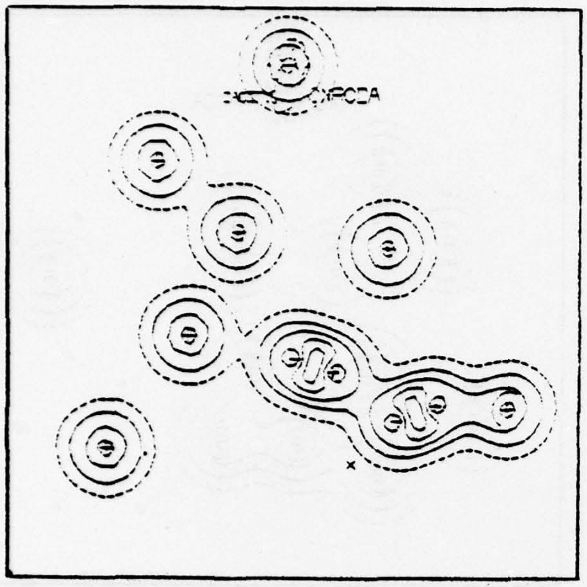
DP Problem 2



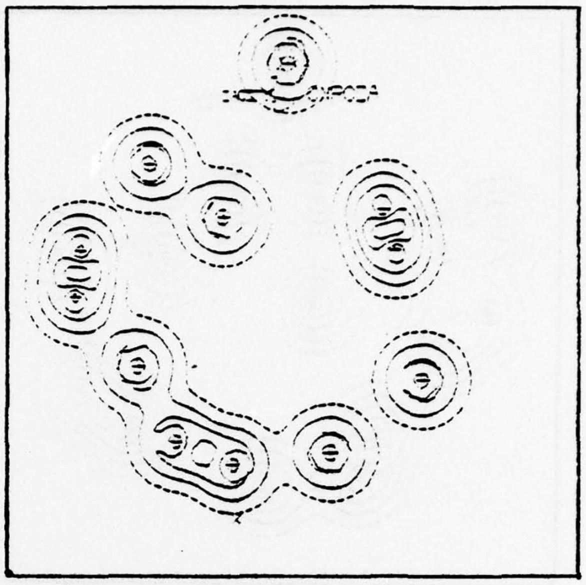
DP Problem 3



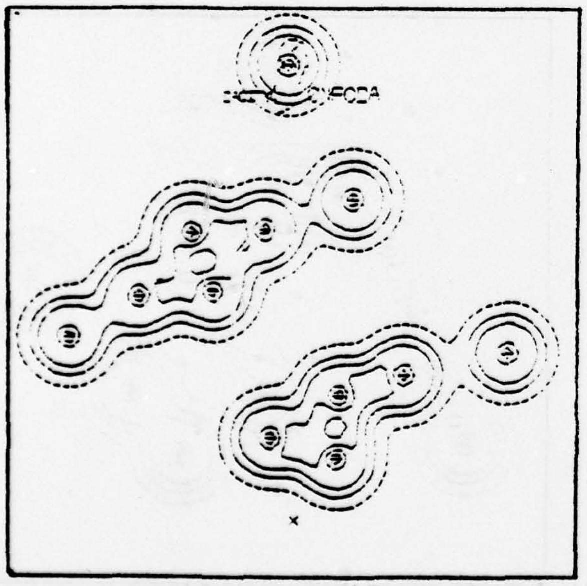
DP Problem 4



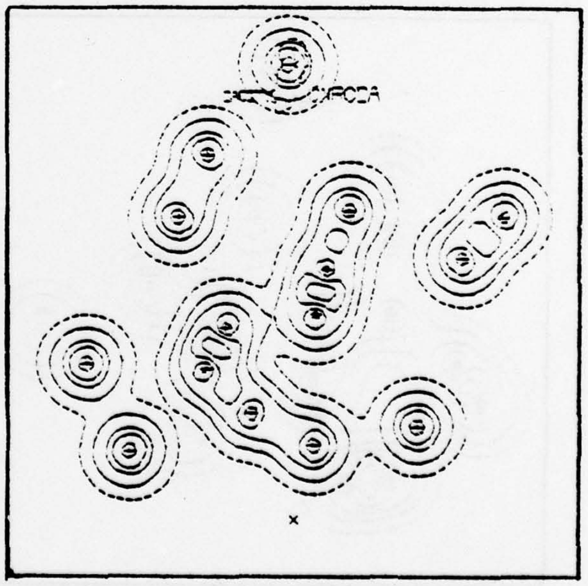
DP Problem 5



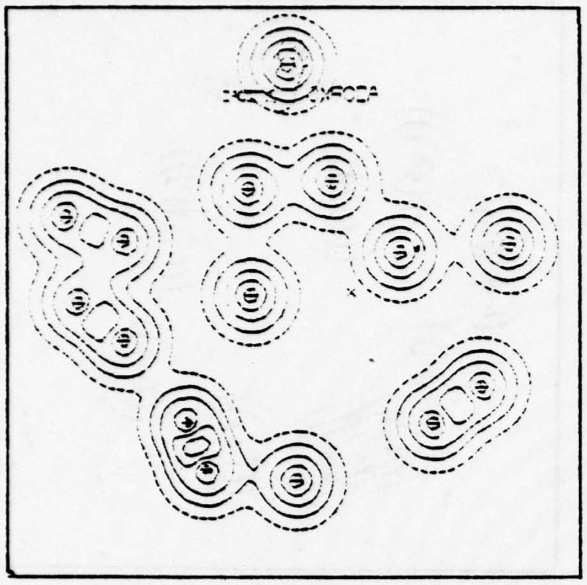
DP Problem 6

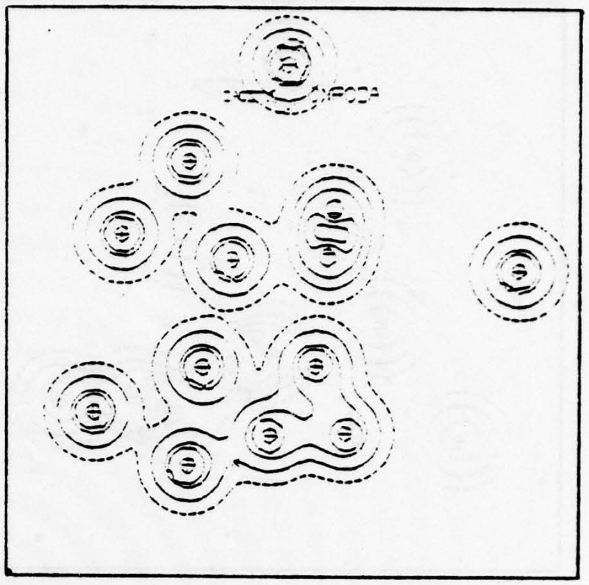


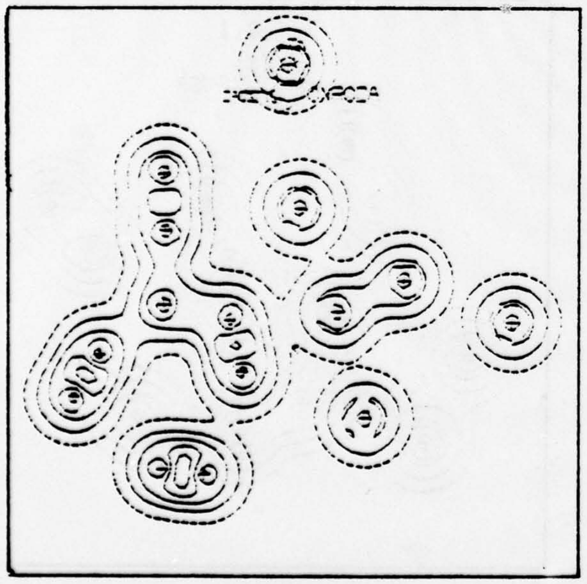
DP Problem 7

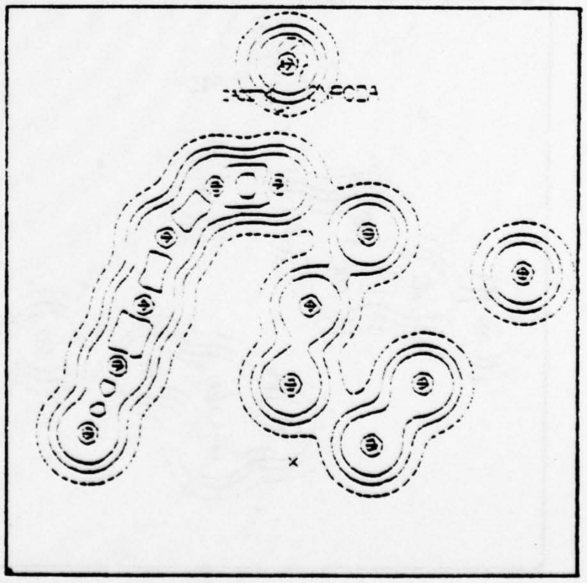


DP Problem 8


DP Problem 9


DP Problem 10


DP Problem 11


DP Problem 12

DISTRIBUTION

Organization	No. of Copies
Director, Engineering Psychology Programs, Code 455 Office of Naval Research 800 North Quincy Street Arlington, Virginia 22217	5
Defense Documentation Center Cameron Station Alexandria, Virginia 22314	12
Col. Henry L. Taylor, USAF OAD (E&LS) ODDR&E Pentagon, Room 3D129 Washington, D. C. 20301	1
Capt. Roger Granum Office of Assistant Secretary of Defense (Intelligence), Pentagon Washington, D. C. 20401	1
Dr. Robert Young Director Cybernetics Technology Office Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia 22209	1
Personnel Logistics Plans, OP987H Office of the Chief of Naval Operations Department of the Navy Washington, D. C. 20350	1
Commanding Officer Office of Naval Research Branch Office Attn: Dr. Charles Davis 536 South Clark Street Chicago, Illinois 60605	1
Dr. A. L. Slafkosky Scientific Advisor Commandant of the Marine Corps Code RD-1 Washington, D. C. 20380	1

Organization	No. of Copies
Assistant Chief for Technology, Code 200 Office of Naval Research 800 North Quincy Street Arlington, Virginia 22217	1
Analysis and Support Division, Code 230 Office of Naval Research 800 North Quincy Street Arlington, Virginia 22217	1
Naval Analysis Programs, Code 431 Office of Naval Research 800 North Quincy Street Arlington, Virginia 22217	1
Operations Research Program, Code 434 Office of Naval Research 800 North Quincy Street Arlington, Virginia 22217	1
Statistics and Probability Program, Code 436 Office of Naval Research 800 North Quincy Street Arlington, Virginia 22217	1
Information Systems Program, Code 437 Office of Naval Research 800 North Quincy Street Arlington, Virginia 22217	1
Commanding Officer Office of Naval Research Branch Office Attn: Dr. J. Lester 495 Summer Street Boston, Massachusetts 02210	1
Commanding Officer Office of Naval Research Branch Office Attn: Dr. E. Gloye 1030 East Green Street Pasadena, California 91106	1

Organization	No. of Copies
Commanding Officer Office of Naval Research Branch Office Attn: Mr. R. Lawson 1030 East Green Street Pasadena, California 91106	1
Office of Naval Research, Code 1021P Department of the Navy 800 North Quincy Street Arlington, Virginia 22217	6
Dr. Fred Muckler Manned Systems Design, Code 311 Navy Personnel Research and Development Center San Diego, California 92152	1
LCDR M. O'Bar, Code 305 Navy Personnel Research and Development Center San Diego, California 92152	1
Navy Personnel Research and Development Center Management Support Department, Code 210 San Diego, California 92152	1
Naval Electronics Systems Command Human Factors Engineering Branch, Code 4701 Washington, D. C. 20360	1
Director, Naval Research Laboratory Technical Information Division, Code 2627 Washington, D. C. 20375	6
Mr. Arnold Rubinstein Naval Material Command, NAVMAT 0344 Department of the Navy Washington, D. C. 20360	1
Mr. John Silva Head, Human Factors Division Naval Ocean Systems Center San Diego, California 92152	1

Organization	No. of Copies
Dr. Jesse Orlansky Institute for Defense Analyses 400 Army-Navy Drive Arlington, Virginia 22202	1
Human Factors Department, Code N215 Naval Training Equipment Center Orlando, Florida 32818	1
Dr. Alfred F. Smode Training Analysis & Evaluation Group Naval Training Equipment Center, Code N-OOT Orlando, Florida 32818	1
Dr. Gary Poock Operations Research Department Naval Postgraduate School Monterey, California 92940	1
Dr. Joseph Zeidner Director, Organization and Systems Research Laboratory U. S. Army Research Institute 5001 Eisenhower Avenue Alexandria, Virginia 22333	1
Dr. Donald A. Topmiller Chief, Systems Effectiveness Branch Human Engineering Division Wright Patterson AFB, Ohio 45433	1
Dr. H. W. Sinaiko Smithsonian Institution 801 N. Pitt Street Alexandria, Virginia 22314	1
Captain Robert Ammann OP-942C Office of the Chief of Naval Operations Washington, D. C. 20350	1
Mr. L. A. Aarons R&D Plans Division Office of the Chief of Naval Operations OP-987C Washington, D. C. 20350	1

Organization	No. of Copies
Commander, Naval Electronics Systems Command Command and Control Division, Code 530 Washington, D. C. 20360	1
Commander, Naval Electronics Systems Command C^3 Project Office PME 108-1 Attn: G. Hamilton Washington, D. C. 20360	1
LCDR Charles Theisen Naval Air Development Center, Code 4024 Warminster, Pennsylvania 18974	1
Dr. S. D. Epstein Analytics 2500 Maryland Road Willow Grove, Pennsylvania 19090	1
Mr. Harold Crane CTEC, Inc. 7777 Leesburg Pike Falls Church, Virginia 22043	1
Dr. Edgar Johnson Organizations & Systems Research Laboratory U. S. Army Research Laboratory 5001 Eisenhower Avenue Alexandria, Virginia 22333	1
Lt. Col. David Dianich HQS Tactical Air Command Langley AFB, Virginia 22065	1
Dr. C. Peterson Decisions and Designs, Inc. 8400 Westpark Drive, Suite 600 McLean, Virginia 22101	1
Mr. George Pugh Decision-Science Applications, Inc. 1401 Wilson Boulevard Arlington, Virginia 22209	1

Organization	No. of Copies
Dr. Amos Freedy Perceptronics, Inc. 6271 Variel Avenue Woodland Hills, California 91364	1
Dr, Miley Merkhofer Stanford Research Institute Decision Analysis Group Menlo Park, California 94025	1
Mr. Robert Garnero Stanford Research Institute Naval Warfare Research Center Menlo Park, California 94025	1
Dr. H. L. Morgan University of Pennsylvania Wharton School Philadelphia, Pennsylvania 19174	1
M. L. Metersky NAVAIRDEVCEN, Code 5424 Warminster, Pennsylvania 18974	1
Dr. Clovis Landry Martin Marietta Aerospace Mail Stop 8105 Denver Division P.O. Box 179 Denver, Colorado 80201	1
Mr. Victor Monteleon Naval Ocean Systems Center Code 230 San Diego, California 92152	1
Commander, Naval Electronics Systems Command ELEX-03 Washington, D. C. 20360	1
Dr. John Shore Code 5403 Communications Sceinces Division Naval Research Laboratory Washington, D. C. 20375	1
Dr. Meredith Crawford 5606 Montgomery Street Chevy Chase, Maryland 20015	1

Organization	No. of Copies
CDR Richard Schlaff NIPSSA Hoffman Building #1 2461 Eisenhower Avenue Alexandria, Virginia 22331	1
Dr. Chantee Lewis Management Department Naval War College Newport, Rhode Island 02840	1
Dr. Arthur Siegel Applied Psychological Services Science Center 404 E. Lancaster Street Wayne, Pennsylvania 19807	1
Mr. Robert Brandenburg ACCAT Naval Ocean Systems Center San Diego, California	1
Lt. Vincent Brackett, Code 83 COMOPTEVFOR Norfold, Virginia 23511	1